PTQ-MCM

Quantum / Unity Platform

Modbus Communication Module

Application Reference Guide

April 29, 2008

Please Read This Notice

Successful application of this module requires a reasonable working knowledge of the Schneider Electric Quantum / Unity hardware, the PTQ-MCM Module and the application in which the combination is to be used. For this reason, it is important that those responsible for implementation satisfy themselves that the combination will meet the needs of the application without exposing personnel or equipment to unsafe or inappropriate working conditions.

This manual is provided to assist the user. Every attempt has been made to ensure that the information provided is accurate and a true reflection of the product's installation requirements. In order to ensure a complete understanding of the operation of the product, the user should read all applicable Schneider Electric documentation on the operation of the Schneider Electric hardware.

Under no conditions will ProSoft Technology be responsible or liable for indirect or consequential damages resulting from the use or application of the product.

Reproduction of the contents of this manual, in whole or in part, without written permission from ProSoft Technology is prohibited.

Information in this manual is subject to change without notice and does not represent a commitment on the part of ProSoft Technology Improvements and/or changes in this manual or the product may be made at any time. These changes will be made periodically to correct technical inaccuracies or typographical errors.

PTQ Installation and Operating Instructions

The statement "power, input and output (I/O) wiring must be in accordance with Class I, Division 2 wiring methods Article 501-10(b) of the National Electrical Code, NFPA 70 for installations in the U.S., or as specified in section 18-1J2 of the Canadian Electrical Code for installations within Canada and in accordance with the authority having jurisdiction".

The following or equivalent warnings shall be included:

A Warning - Explosion Hazard - Substitution of components may Impair Suitability for Class I, Division 2;
B Warning - Explosion Hazard - When in Hazardous Locations, Turn off Power before replacing Wiring Modules, and
C Warning - Explosion Hazard - Do not Disconnect Equipment unless Power has been switched Off or the Area is known to be Nonhazardous.
D Caution: The Cell used in this Device may Present a Fire or Chemical Burn Hazard if Mistreated. Do not Disassemble, Heat above 100°C (212°F) or Incinerate.

Important Notice:

CAUTION: THE CELL USED IN THIS DEVICE MAY PRESENT A FIRE OR CHEMICAL BURN HAZARD IF MISTREATED. DO NOT DISASSEMBLE, HEAT ABOVE 100°C (212°F) OR INCINERATE.

Maximum battery load = 200 µA.

Maximum battery charge voltage = 3.4 VDC.

Maximum battery charge current = 500 µA.

Maximum battery discharge current = 30 µA.

Your Feedback Please

We always want you to feel that you made the right decision to use our products. If you have suggestions, comments, compliments or complaints about the product, documentation or support, please write or call us.

ProSoft Technology
1675 Chester Avenue, Fourth Floor
Bakersfield, CA 93301
+1 (661) 716-5100
+1 (661) 716-5101 (Fax)
http://www.prosoft-technology.com

Copyright © ProSoft Technology, Inc. 2000 - 2008. All Rights Reserved.

PTQ-MCM Application Reference Guide
April 29, 2008
PSFT..PTQ.UM.08.04.29

ProSoft Technology ®, ProLinx ®, inRAx ®, ProTalk® and RadioLinx ® are Registered Trademarks of ProSoft Technology, Inc.

Contents

PLEASE READ THIS NOTICE **2**
PTQ Installation and Operating Instructions 2
Important Notice: 2
Your Feedback Please 3
GUIDE TO THE PTQ-MCM APPLICATION REFERENCE GUIDE **9**
1 START HERE **11**
1.1 Hardware and Software Requirements 11
1.1.1 ProTalk Module Carton Contents 11
1.1.2 Quantum / Unity Hardware 12
1.1.3 PC and PC Software 12
1.2 Install ProSoft Configuration Builder Software 13
2 CONFIGURING THE PROCESSOR WITH CONCEPT **15**
2.1 Information for Concept Version 2.6 Users 16
2.1.1 Installing MDC Configuration Files 16
2.2 Create a New Project 17
2.3 Add the PTQ Module to the Project 20
2.4 Set up Data Memory in Project 22
2.5 Download the Project to the Processor 25
2.6 Verify Successful Download 28
3 CONFIGURING THE PROCESSOR WITH PROWORX **33**
4 CONFIGURING THE PROCESSOR WITH UNITYPRO XL **37**
4.1 Create a New Project 37
4.2 Add the PTQ Module to the Project 39
4.3 Build the Project 41
4.4 Connect Your PC to the Processor 42
4.4.1 Connecting to the Processor with TCPIP 44
4.5 Download the Project to the Processor 44
5 SETTING UP THE PROTALK MODULE **47**
5.1 Install the ProTalk Module in the Quantum Rack 47
5.1.1 Verify Jumper Settings 47
5.1.2 Inserting the 1454-9F connector 48
5.1.3 Install the ProTalk Module in the Quantum Rack 48
5.1.4 Cable Connections 49
5.2 Connect the PC to the ProTalk Configuration/Debug Port 53
6 MODIFYING THE CONFIGURATION FILE **57**
6.1 ProSoft Configuration Builder 57
6.1.1 Set Up the Project 58
6.1.2 Set Module Parameters 60
6.2 Edit the Configuration File 62
6.2.1 [Module] 63
6.2.2 [Backplane Configuration] 63
6.2.3 [MCM Port X] 65
6.2.4 [Modbus Port 1 Commands] 70
6.3 Download the Project to the Module 76
6.4 Verification and Troubleshooting 77
7 DIAGNOSTICS AND TROUBLESHOOTING **79**
7.1 Reading Status Data from the Module 79
7.1.1 Required Hardware 79

7.1.2 The Configuration/Debug Menu ... 80
7.1.3 Main Menu ... 82
7.1.4 Modbus Database View ... 84
7.1.5 Backplane Menu ... 86
7.1.6 Data Analyzer ... 87
7.1.7 Data Analyzer Tips ... 89
7.1.8 Protocol Serial Menu ... 91
7.1.9 Master Command Error List Menu ... 93
7.2 LED Status Indicators ... 94
7.3 Error Status Table ... 95
7.3.1 Standard Modbus Protocol Errors ... 95
7.3.2 Module Communication Error Codes ... 95
7.3.3 Command List Entry Errors ... 95
8 REFERENCE ... 97
8.1 Product Specifications ... 97
8.1.1 Features and Benefits ... 97
8.1.2 General Specifications ... 97
8.1.3 Hardware Specifications ... 98
8.1.4 Functional Specifications ... 98
8.2 Functional Overview ... 99
8.2.1 Backplane Data Transfer ... 99
8.2.2 Special Functions ... 100
8.2.3 Event Command ... 100
8.2.4 Command Control Block ... 101
8.2.5 Warm Boot ... 104
8.2.6 Cold Boot ... 104
8.2.7 Pass-Thru Control Blocks ... 105
8.3 Status Data Definition ... 110
8.3.1 Status Data Block Structure ... 110
8.4 Configuration Data ... 112
8.4.1 Port 1 Setup ... 112
8.4.2 Port 2 Setup ... 113
8.4.3 Port 1 Commands ... 113
8.4.4 Port 2 Commands ... 114
8.5 Port Specific Information ... 114
8.5.1 Serial Port Details ... 114
8.5.2 RS-232 Configuration/Debug Port ... 114
8.6 Modbus Protocol Specification ... 114
8.6.1 Read Output Status (Function Code 01) ... 114
8.6.2 Read Input Status (Function Code 02) ... 115
8.6.3 Read Holding Registers (Function Code 03) ... 116
8.6.4 Read Input Registers (Function Code 04) ... 117
8.6.5 Force Single Coil (Function Code 05) ... 118
8.6.6 Preset Single Register (Function Code 06) ... 119
8.6.7 Diagnostics (Function Code 08) ... 119
8.6.8 Force Multiple Coils (Function Code 15) ... 122
8.6.9 Preset Multiple Registers (Function Code 16) ... 122
8.6.10 MODBUS Exception Responses ... 123
8.7 Frequently Asked Questions ... 125
8.7.1 What kind of data transfer rates can I expect between the PLC and the module? ... 125
8.7.2 Is a .MDC available for configuration of the Module? ... 125
8.7.3 Does the module work in a remote rack? ... 125
8.7.4 Can I use the module in a hot backup system? ... 125
9 SUPPORT, SERVICE & WARRANTY ... 127
9.1 How to Contact Us: Sales and Support ... 128
9.2 Return Material Authorization (RMA) Policies and Conditions ... 128
9.2.1 All Product Returns ... 129
9.3 Procedures for Return of Units Under Warranty ... 129
9.4 Procedures for Return of Units Out of Warranty ... 130

9.4.1 Un-repairable Units 130
9.5 LIMITED WARRANTY 131
9.5.1 What Is Covered By This Warranty 131
9.5.2 What Is Not Covered By This Warranty 132
9.5.3 DISCLAIMER REGARDING HIGH RISK ACTIVITIES 133
9.5.4 DISCLAIMER OF ALL OTHER WARRANTIES 133
9.5.5 LIMITATION OF REMEDIES** 133
9.5.6 Time Limit for Bringing Suit 133
9.5.7 No Other Warranties 134
9.5.8 Intellectual Property 134
9.5.9 Additional Restrictions Relating To Software And Other Intellectual Property 134
9.5.10 Allocation of risks 134
9.5.11 Controlling Law and Severability 135
10 INDEX 137

Guide to the PTQ-MCM Application Reference Guide

Function		Section to Read	Details
Introduction (Must Do)	→	Start Here (page 11)	This Section introduces the customer to the module. Included are: package contents, system requirements, hardware installation, and basic configuration.
Verify Communication, Diagnostic and Troubleshooting	→	Verifying Communication (page 77) Diagnostics and Troubleshooting (page 79)	This section describes how to verify communications with the network. Diagnostic and Troubleshooting procedures.
Reference Product Specifications Functional Overview Glossary	→	Reference (page 97) Functional Overview (page 99) Product Specifications (page 97)	These sections contain general references associated with this product, Specifications, and the Functional Overview.
Support, Service, and Warranty Index	→	Support, Service and Warranty (page 127)	This section contains Support, Service and Warranty information. Index of chapters.

1 Start Here

In This Chapter

- Hardware and Software Requirements 11
- Install ProSoft Configuration Builder Software..... 13

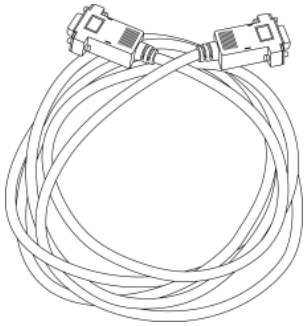
This guide is intended to guide you through the ProTalk module setup process, from removing the module from the box to exchanging data with the processor. In doing this, you will learn how to:

- Set up the processor environment for the PTQ module
- View how the PTQ module exchanges data with the processor
- Edit and download configuration files from your PC to the PTQ module
- Monitor the operation of the PTQ module

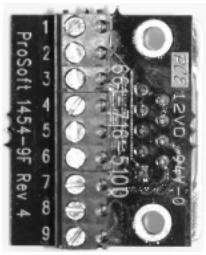
1.1 Hardware and Software Requirements

1.1.1 ProTalk Module Carton Contents

ProTalk Module



Null Modem Serial Cable



1454-9F DB-9 Female to 9 Pos Screw Terminal adapter (Serial protocol modules only)	ProSoft Solutions CD

Note: The DB-9 Female to 5 Pos Screw Terminal adapter is not required on Ethernet modules and is therefore not included in the carton with these types of modules.

1.1.2 Quantum / Unity Hardware

This guide assumes that you are familiar with the installation and setup of the Quantum / Unity hardware. The following should be installed, configured and powered up before proceeding:

- Quantum or Unity Processor
- Quantum rack
- Quantum power supply
- Quantum Modbus Plus Network Option Module (NOM Module) (optional)
- Quantum to PC programming hardware
- NOM Ethernet or Serial connection to PC

1.1.3 PC and PC Software

- Windows-based PC with at least one COM port
- Quantum programming software installed on machine

  or

- Concept™ PLC Programming Software version 2.6

  or
  ProWORX PLC Programming Software
  or
  UnityPro XL PLC Programming Software

- HyperTerminal (used in this guide) This is a communication program that is included with Microsoft Windows. You can normally find it in **Start / Programs / accessories / Communications**.

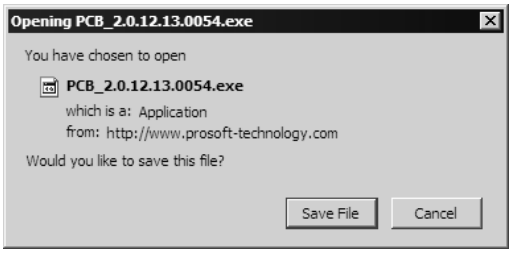
Note: ProTalk modules are compatible with common Quantum / Unity programming applications, including Concept and UnityPro XL. For all other programming applications, please contact technical support.

1.2 Install ProSoft Configuration Builder Software

You must install the ProSoft Configuration Builder (PCB) software in order to configure the PTQ-MCM module. You can always get the newest version of ProSoft Configuration Builder from the ProSoft Technology web site.

To install ProSoft Configuration Builder from the ProSoft Web Site

1. Open your web browser and navigate to http://www.prosoft-technology.com/pcb
2. Click the **Download Here** link to download the latest version of ProSoft Configuration Builder.
3. Choose "Save" or "Save File" when prompted. The following illustrations show the file download prompt for two of the most common web browsers.



File Download - Security Warning
Do you want to run or save this file?
Name: PCB_2.0.12.13.0054.exe
Type: Application, 17.3MB
From: www.prosoft-technology.com
Run
Save
Cancel
While files from the Internet can be useful, this file type can potentially harm your computer. If you do not trust the source, do not run or save this software. What's the risk?

4. Make a note of the location where you saved the file, for example "Desktop", or "My Documents", so you can start the installation program.
5. When the download is complete, locate and open the file, and then follow the instructions on your screen to install the program.

If you do not have access to the Internet, you can install ProSoft Configuration Builder from the ProSoft Solutions CD-ROM, included in the package with your PTQ-MCM module.

To install ProSoft Configuration Builder from the CD-ROM

1. Insert the ProSoft Solutions CD-ROM into the CD drive of your PC. Wait for the startup screen to appear.
2. On the startup screen, click *Product Documentation*. This action opens an explorer window.

3 Click to open the *Utilities* folder. This folder contains all of the applications and files you will need to set up and configure your module.
4 Double-click the *ProSoft Configuration Builder Setup* program and follow the instructions on your screen to install the software on your PC.

Note: Many of the configuration and maintenance procedures use files and other utilities on the CD-ROM. You may wish to copy the files from the Utilities folder on the CD-ROM to a convenient location on your hard drive.

2 Configuring the Processor with Concept

In This Chapter

- Information for Concept Version 2.6 Users 16
- Create a New Project 17
- Add the PTQ Module to the Project 20
- Set up Data Memory in Project 22
- Download the Project to the Processor 25
- Verify Successful Download 28

The following steps are designed to ensure that the processor is able to transfer data successfully with the PTQ module. As part of this procedure, you will use Concept configuration software from Schneider Electric to create a project, add the PTQ module to the project, set up data memory for the project, and then download the project to the processor.

Important Note: Concept software does not report whether the PTQ module is present in the rack, and therefore is not able to report the health status of the module when the module is online with the Quantum processor. Please take this into account when monitoring the status of the PTQ module.

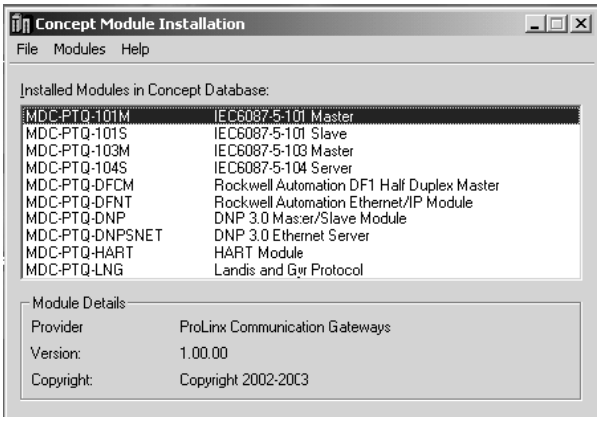
2.1 Information for Concept Version 2.6 Users

This guide uses Concept PLC Programming Software version 2.6 to configure the Quantum PLC. The ProTalk installation CD includes MDC module configuration files that help document the PTQ installation. Although not required, these files should be installed before proceeding to the next section.

2.1.1 Installing MDC Configuration Files

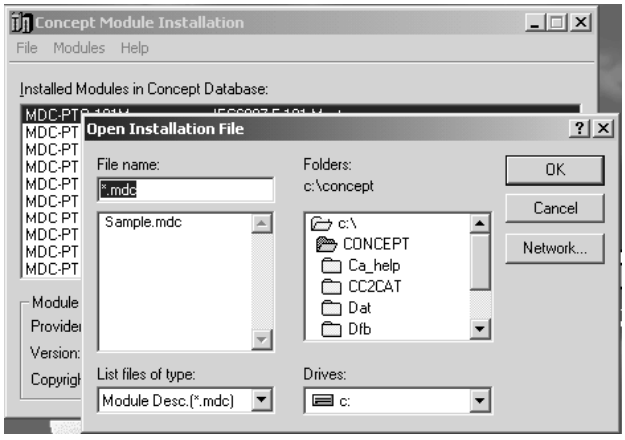
1 From a PC with Concept 2.6 installed, choose **Start / Programs / Concept / ModConnect Tool**.

This action opens the Concept Module Installation dialog box.



2 Choose **File / Open Installation File**.

This action opens the Open Installation File dialog box:

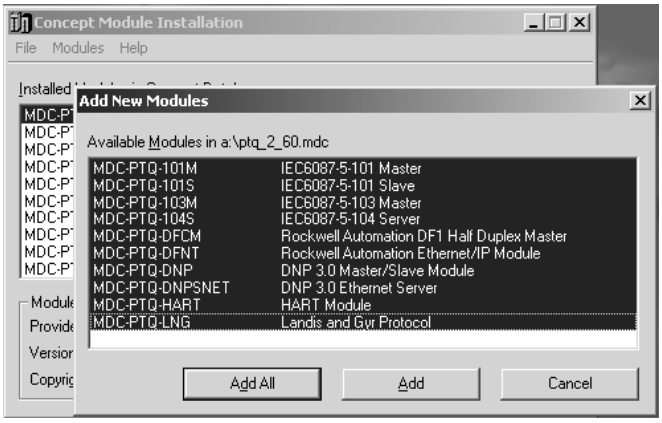


3 If you are using a Quantum processor, you will need the MDC files. In the Open Installation File dialog box, navigate to the **MDC Files** directory on the ProTalk CD.

4 Choose the MDC file and help file for your version of Concept:

- Concept 2.6 users: select PTQ_2_60.mdc and PTQMDC.hlp
- Concept 2.5 users: select PTQ_2_50.mdc and PTQMDC.hlp.

Select the files that go with the Concept version you are using, and then click **OK**. This action opens the add New Modules dialog box.

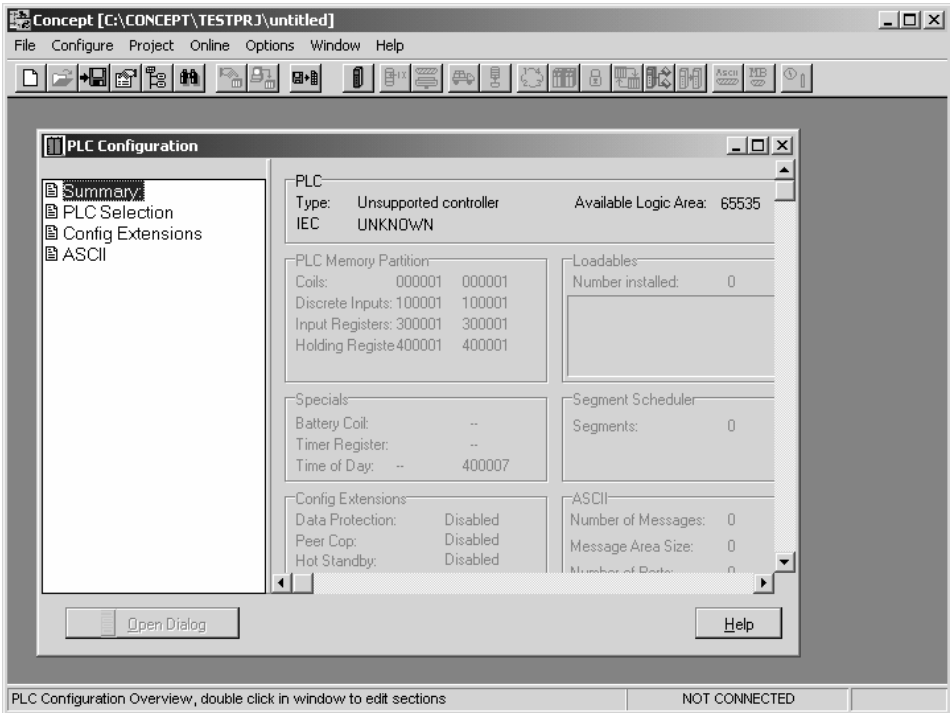


5 Click the **add all** button. A series of message boxes may appear during this process. Click **Yes** or **OK** for each message that appears.
6 When the process is complete, open the File menu and choose Exit to save your changes.

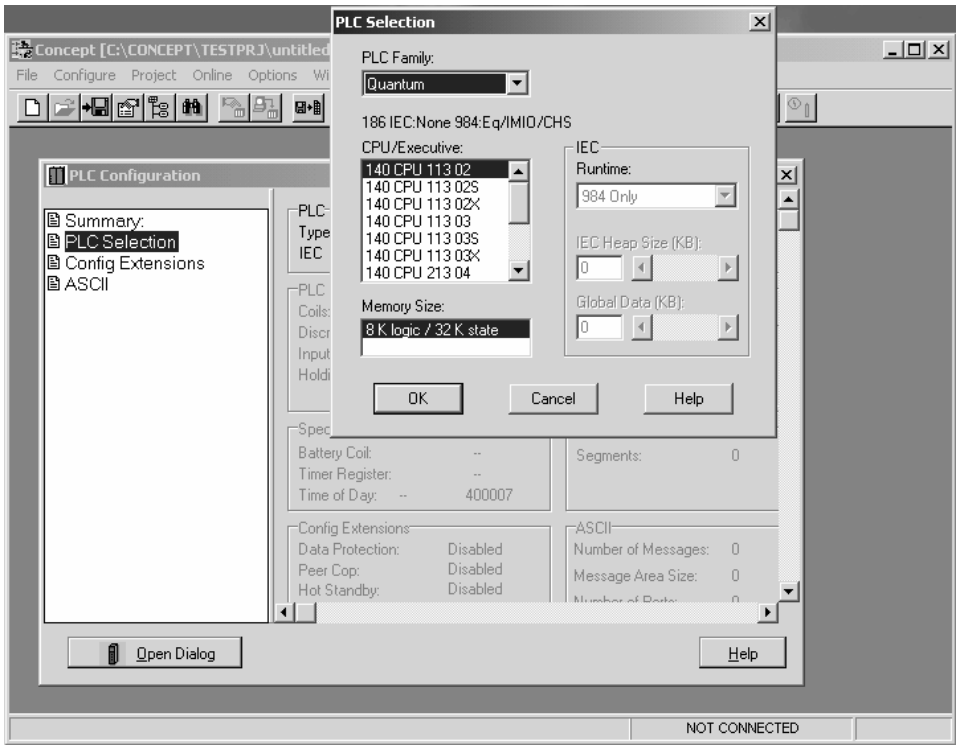
2.2 Create a New Project

This phase of the setup procedure must be performed on a computer that has the Concept configuration software installed.

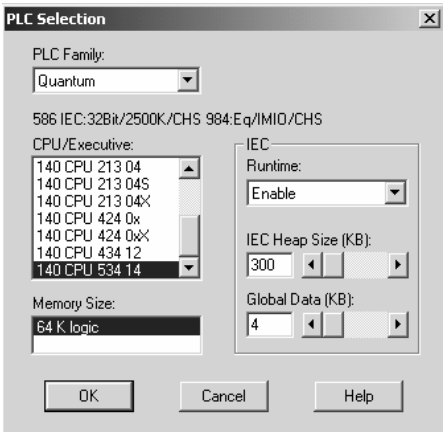
1 From your computer, choose **Start / Programs / Concept V2.6 XL.EN / Concept**. This action opens the Concept window.
2 Open the File menu, and then choose **New Project**. This action opens the PLC Configuration dialog box.



3 In the list of options on the left side of this dialog box, double-click the *PLC Selection* folder. This action opens the PLC Selection dialog box.

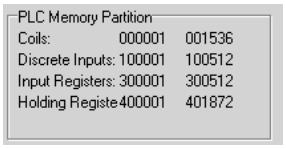


4 In the *CPU/Executive* pane, use the scroll bar to locate and select the PLC to configure.



5 Click **OK**. This action opens the *PLC Configuration* dialog box, populated with the correct values for the PLC you selected.

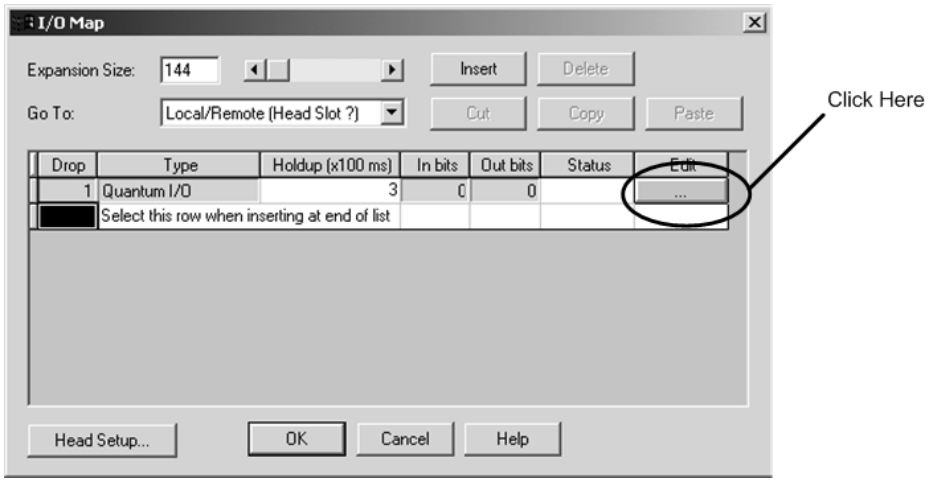
6 Make a note of the holding registers for the module. You will need this information when you modify your application as outlined in the ProTalk application Reference Guides. The Holding Registers are displayed in the PLC Memory Partition pane of the PLC Configuration dialog box.



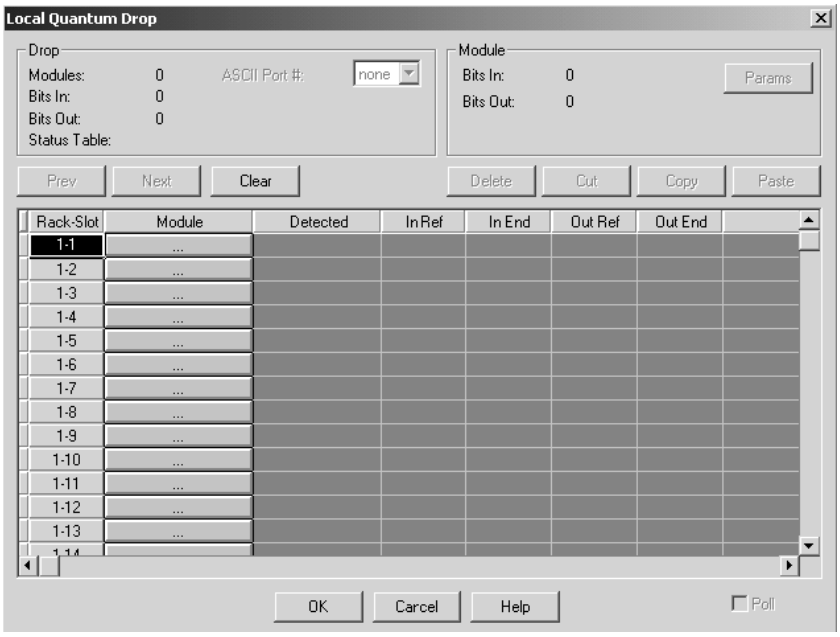
2.3 Add the PTQ Module to the Project

The next step is to add one or more of the PTQ modules to the Project. To add modules:

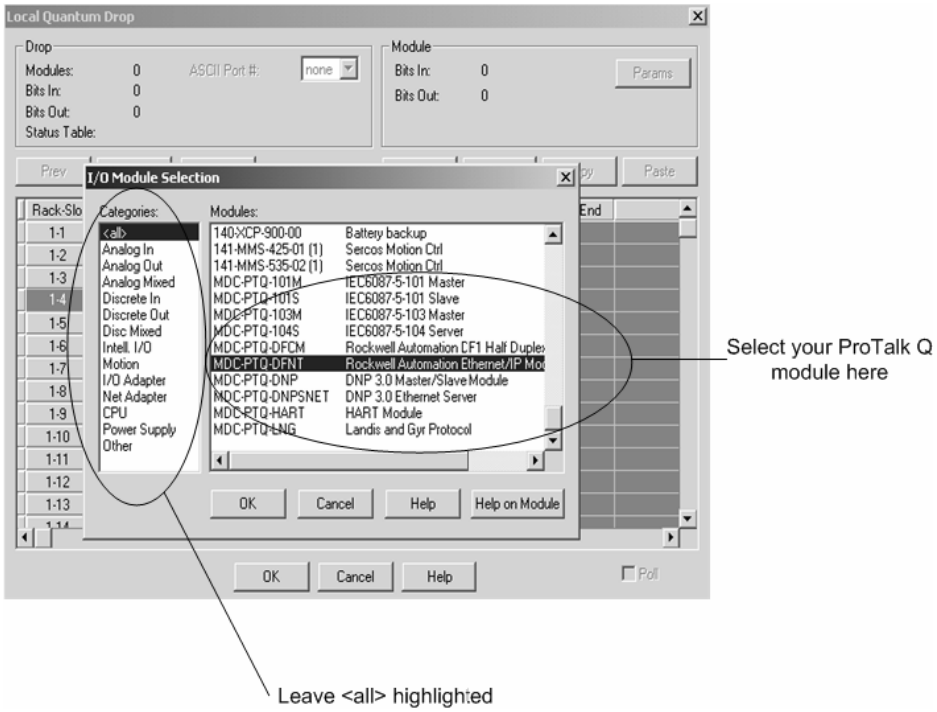
1 In the list of options on the left side of the *PLC Configuration* dialog box, double-click *I/O Map*. This action opens the I/O Map dialog box.



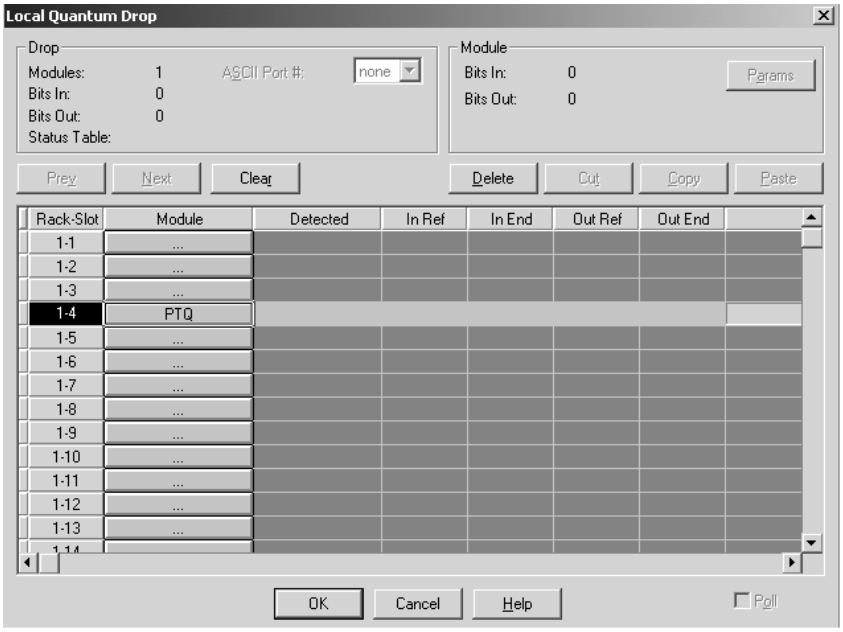
2 Click the **Edit** button to open the *Local Quantum Drop* dialog box. This dialog box is where you identify rack and slot locations.



3 Click the Module button next to the rack/slot position where the ProTalk module will be installed. This action opens the I/O Module Selection dialog box.



4 In the Modules pane, use the scroll bar to locate and select the ProTalk module, and then click OK. This action copies the description of the ProTalk module next to the assigned rack and slot number of the Local Quantum Drop dialog box.



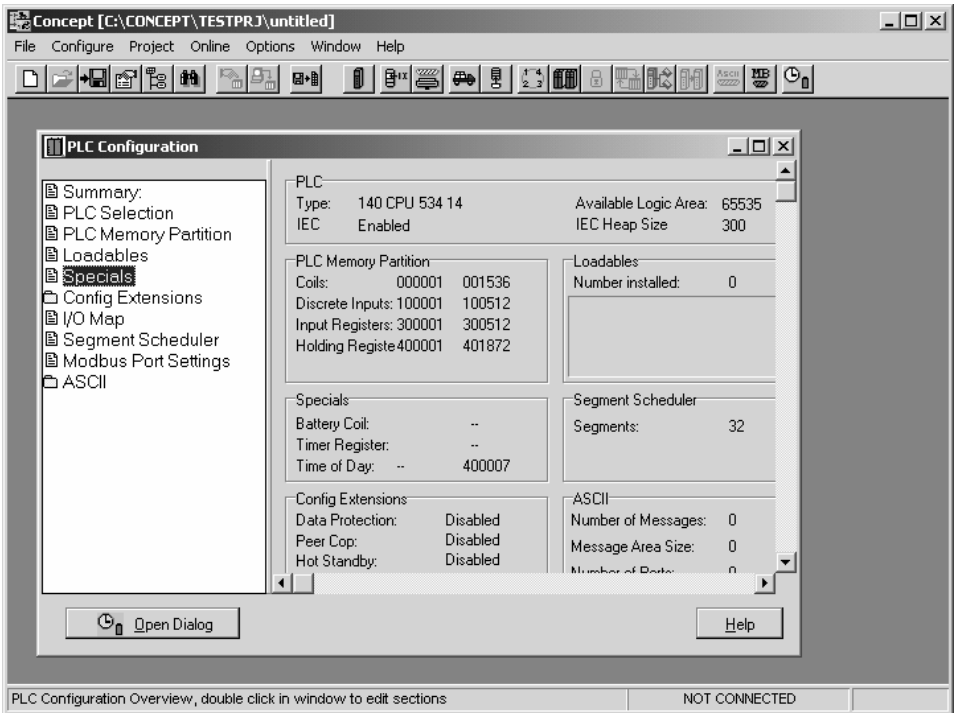
5 Repeat steps 3 through 5 for each ProTalk module you plan to install. When you have finished installing your ProTalk modules, click OK to save your settings. Click Yes to confirm your settings.

Tip: Select a module, and then click the Help on Module button for help pages.

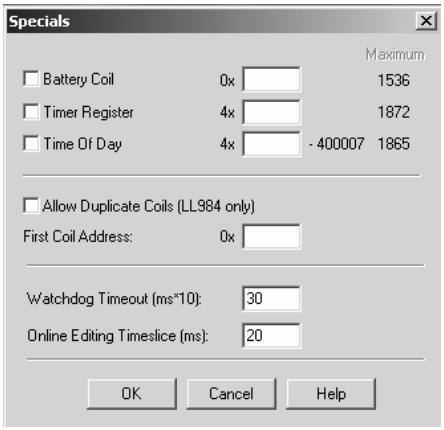
About ProTalk Q
File Edit Bookmark Options Help
Help Topics Back Print Options
PTQ-DNPSNET DNP Server over Ethernet Module
ProTalk Q
The ProTalk Q DNP Server Ethernet Communications Module (PTQ-DNPSNET) supports the implementation of the DNP 3.0 (Distributed Network Protocol) over Ethernet, allowing Quantum processors to easily communicate with host systems supporting the protocol. The module supports DNP Subset Level 2 features and some Level 3 features.
The module supports DNP Subset Level 2 features and some of the Level 3 features allowing the many SCADA systems supporting the DNP protocol to be integrated into the Quantum platform. The module acts as a gateway between the DNP network and the Quantum processor. The data transfer from the Quantum processor is asynchronous from the actions on the DNP network. Databases are user defined and stored in the module to hold the data required by the protocol.
The PTQ-DNPSNET module is a powerful communication interface for Quantum platform processors. Developed under license from Schneider Electric, the module incorporates proprietary backplane technology that enables powerful data access to the Quantum processor.
General Specifications
Functional Specifications
Hardware Specifications
DNP Subset Definition

2.4 Set up Data Memory in Project

1 In the list of options on the left side of the PLC Configuration dialog box, double-click Specials.

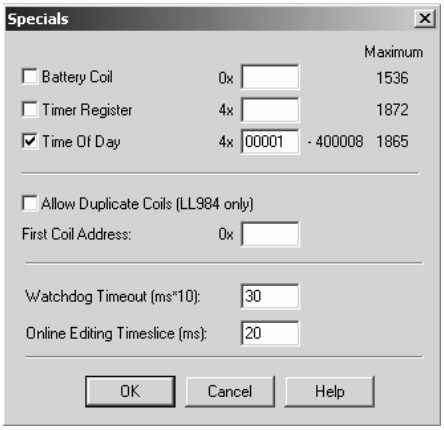


2 This action opens the Specials dialog box.



Selecting the Time of Day

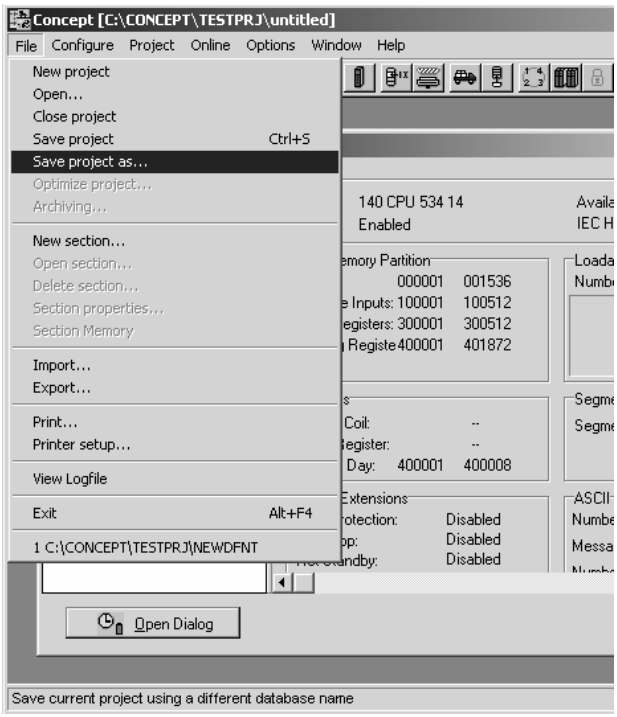
1 Select (check) the Time of Day box, and then enter the value 00001 as shown in the following example. This value sets the first time of day register to 400001.



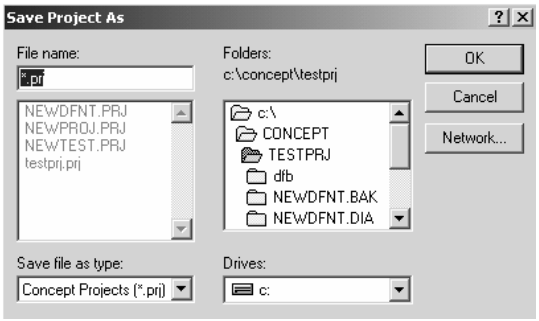
2 Click OK to save your settings and close the Specials dialog box.

Saving your project

1 In the PLC Configuration dialog box, choose File / Save project as.



2 This action opens the Save Project as dialog box.

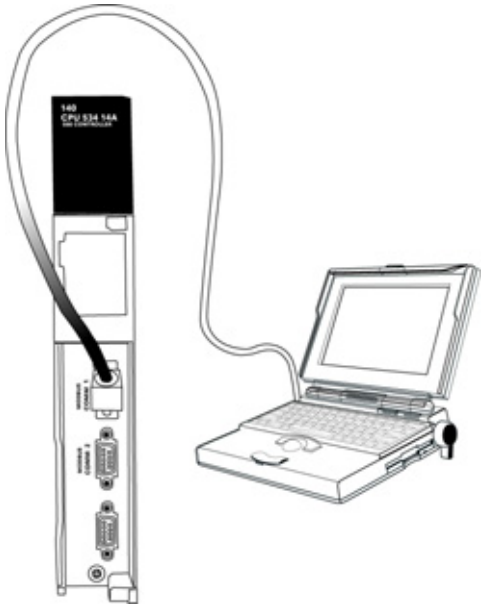


3 Name the project, and then click OK to save the project to a file.

2.5 Download the Project to the Processor

The next step is to download (copy) the project file to the Quantum Processor.

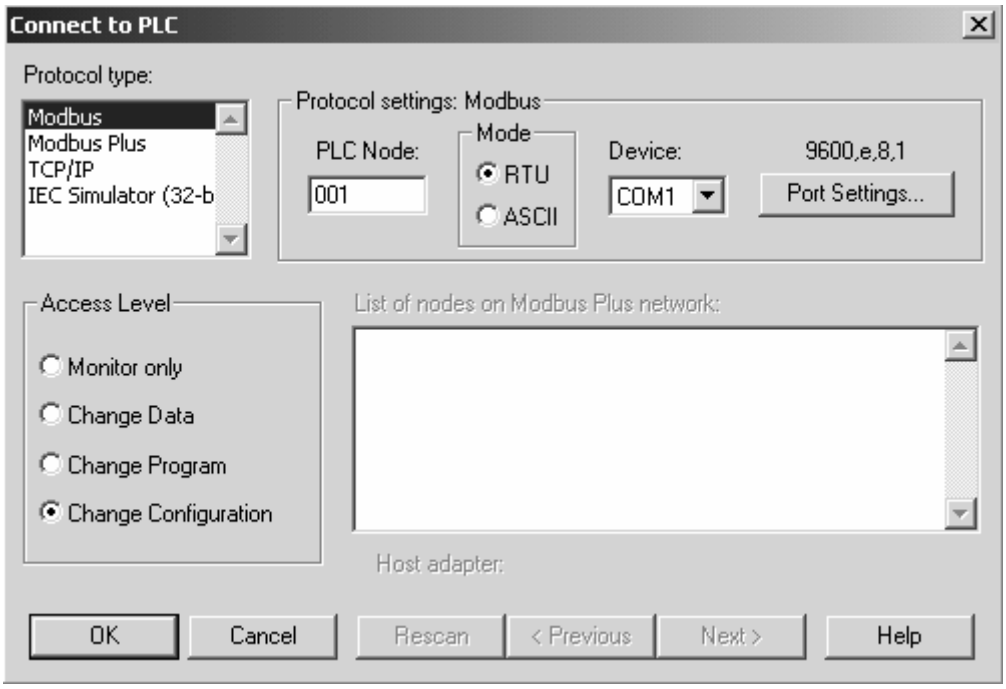
1 Use the null modem cable to connect your PC's serial port to the Quantum processor, as shown in the following illustration.



Note: You can use a Modbus Plus Network Option Module (NOM Module) module in place of the serial port if necessary.

2 Open the PLC menu, and then choose Connect.

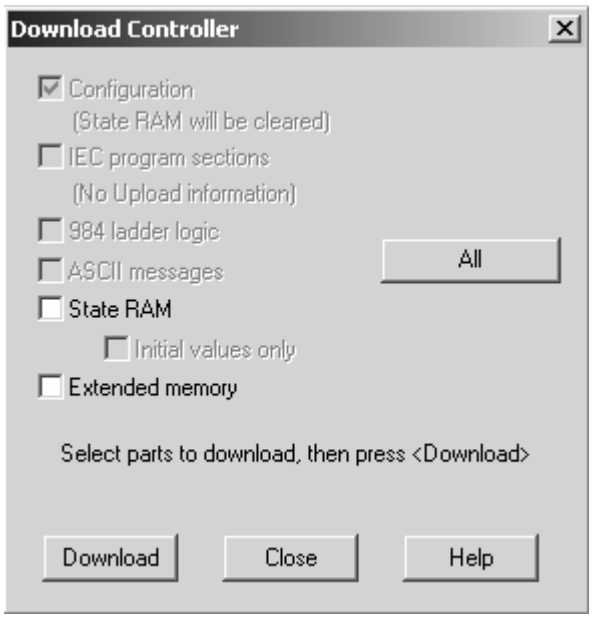
3 In the PLC Configuration dialog box, open the Online menu, and then choose Connect. This action opens the Connect to PLC dialog box.



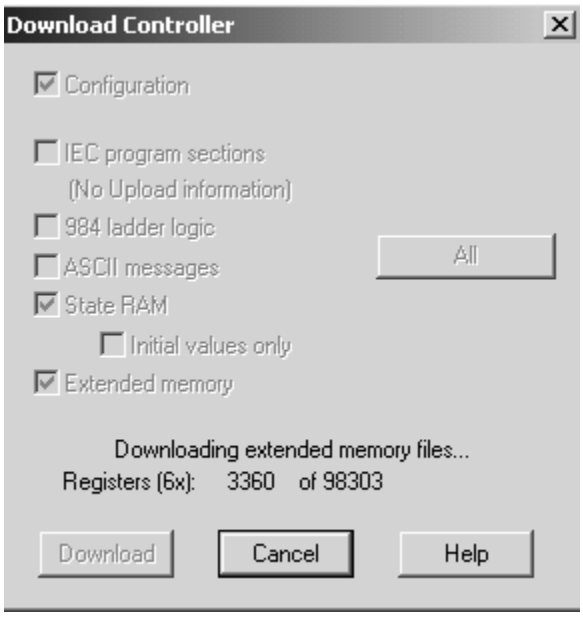
4 Leave the default settings as shown and click OK.

Note: Click OK to dismiss any message boxes that appear during the connection process.

5 In the PLC Configuration window, open the Online menu, and then choose Download. This action opens the Download Controller dialog box.



6 Click all, and then click Download. If a message box appears indicating that the controller is running, click Yes to shut down the controller. The Download Controller dialog box displays the status of the download as shown in the following illustration.

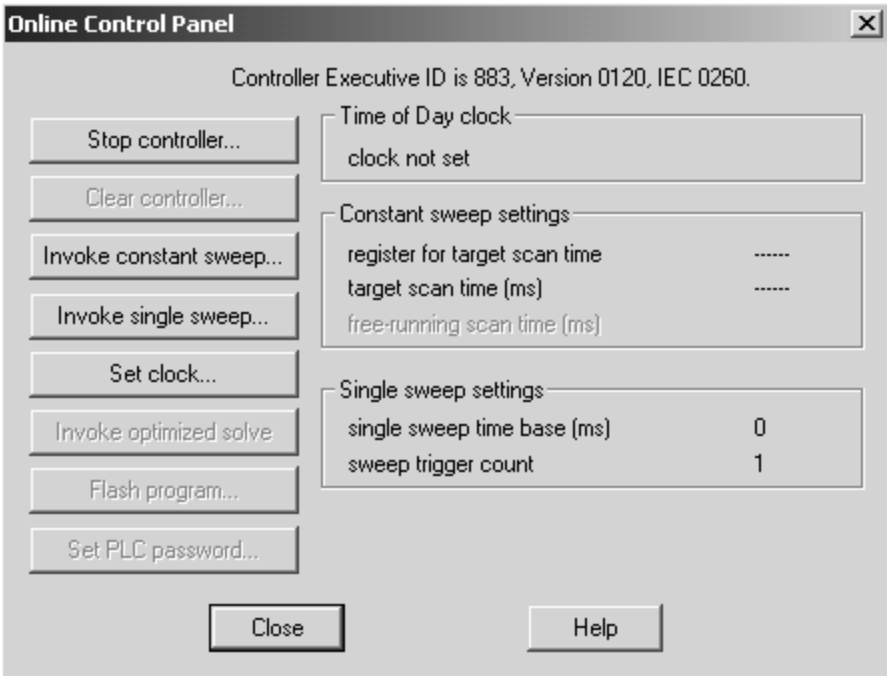


7 When the download is complete, you will be prompted to restart the controller. Click Yes to restart the controller.

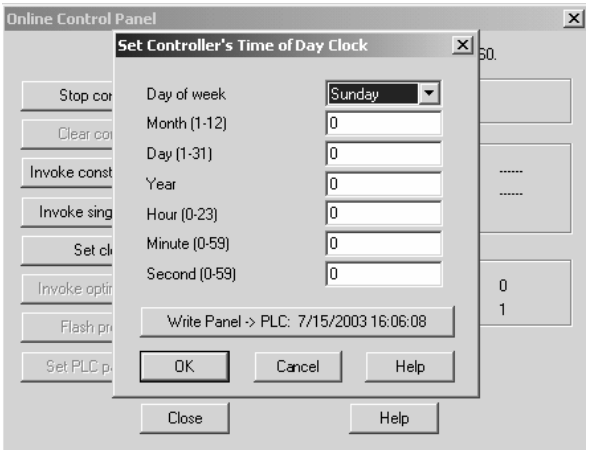
2.6 Verify Successful Download

The final step is to verify that the configuration changes you made were received successfully by the module, and to make some adjustments to your settings.

1 In the PLC Configuration window, open the Online menu, and then choose Online Control Panel. This action opens the Online Control Panel dialog box.



2 Click the Set Clock button to open the Set Controller's Time of Day Clock dialog box.

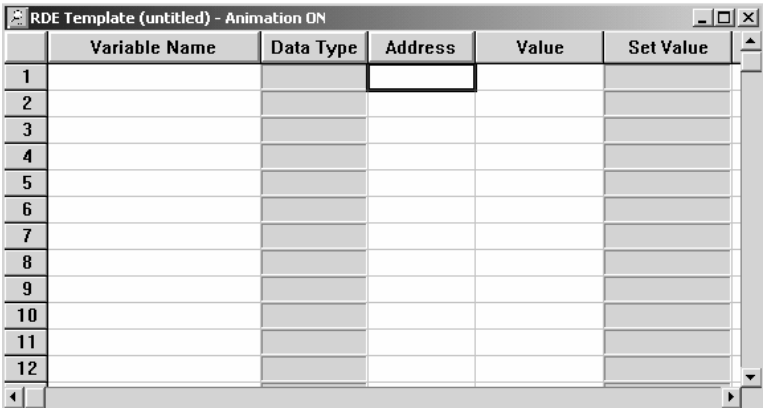


3 Click the Write Panel button. This action updates the date and time fields in this dialog box. Click OK to close this dialog box and return to the previous window.

4 Click Close to close the Online Control Panel dialog box.

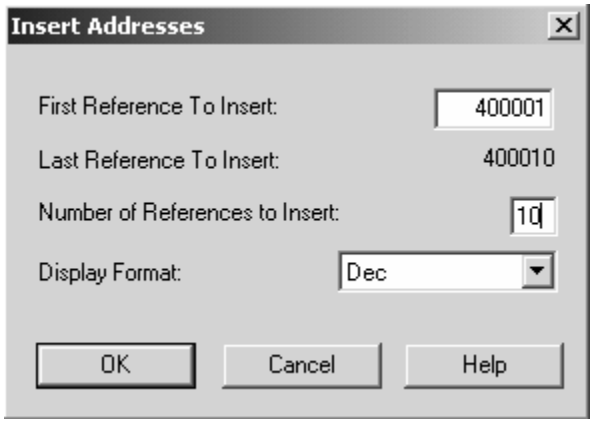
5 In the PLC Configuration window, open the Online menu, and then choose Reference Data Editor. This action opens the Reference Data Editor dialog box. On this dialog box, you will add preset values to data registers that will later be monitored in the ProTalk module.

6 Place the cursor over the first address field, as shown in the following illustration.

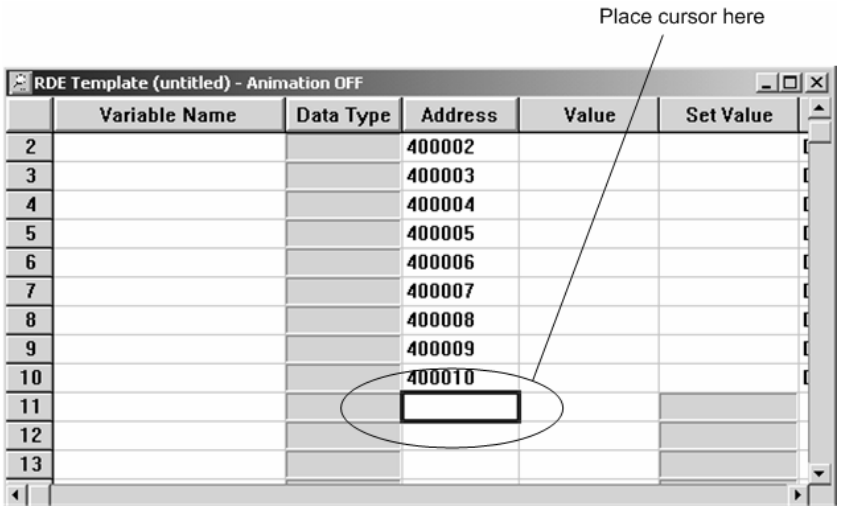


7 In the PLC Configuration window, open the Templates menu, and then choose Insert addresses. This action opens the Insert addresses dialog box.

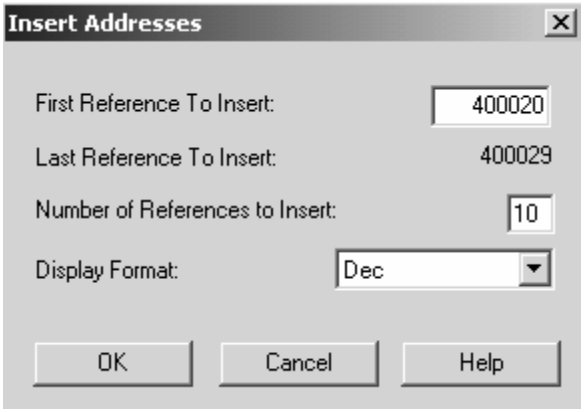
8 On the Insert addresses dialog box, enter the values shown in the following illustration, and then click OK.



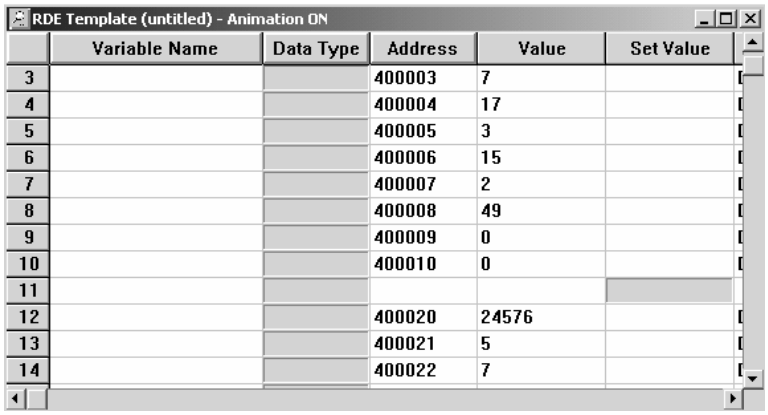
9 Notice that the template populates the address range, as shown in the following illustration. Place your cursor as shown in the first blank address field below the addresses you just entered.



10 Repeat steps 6 through 9, using the values in the following illustration:



11 In the PLC Configuration window, open the Online menu, and then choose animate. This action opens the RDE Template dialog box, with animated values in the Value field.



	Variable Name	Data Type	Address	Value	Set Value
3			400003	7	
4			400004	17	
5			400005	3	
6			400006	15	
7			400007	2	
8			400008	49	
9			400009	0	
10			400010	0	
11					
12			400020	24576	
13			400021	5	
14			400022	7	

12 Verify that values shown are cycling, starting from address 400065 on up.

13 In the PLC Configuration window, open the Templates menu, and then choose Save Template as. Name the template ptqclock, and then click OK to save the template.

14 In the PLC Configuration window, open the Online menu, and then choose Disconnect. At the disconnect message, click Yes to confirm your choice.

At this point, you have successfully

- Created and downloaded a Quantum project to the PLC
- Preset values in data registers that will later be monitored in the ProTalk module.

You are now ready to complete the installation and setup of the ProTalk module.

3 Configuring the Processor with ProWORX

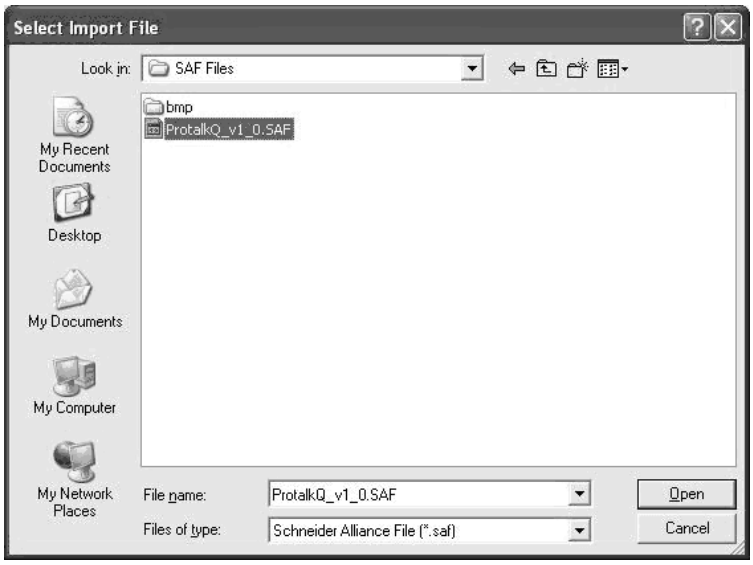
When you use ProWORX 32 software to configure the processor, use the example SaF file provided on the ProTalk Solutions CD-ROM.

Important Note: Proworx software does not report whether the PTQ module is present in the rack, and therefore is not able to report the health status of the module when the module is online with the Quantum processor. Please take this into account when monitoring the status of the PTQ module.

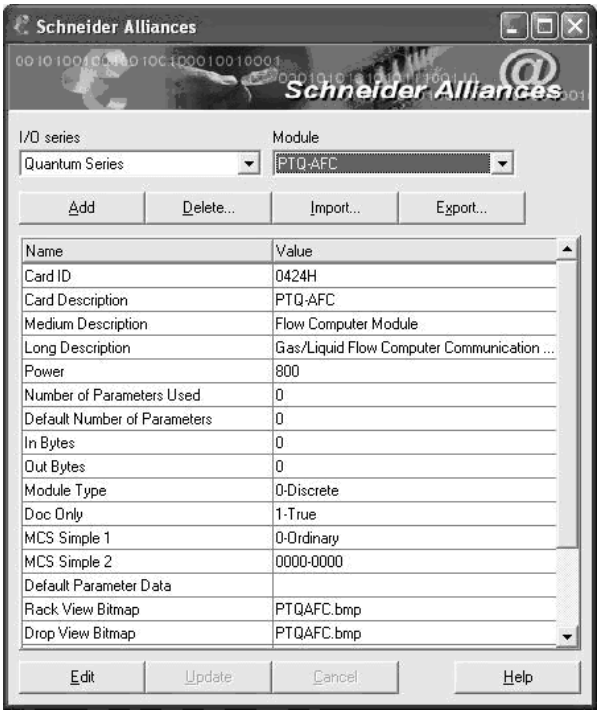
1 Run the Schneider_alliances.exe application that is installed with the Proworx 32 software:

2 Click on Import…

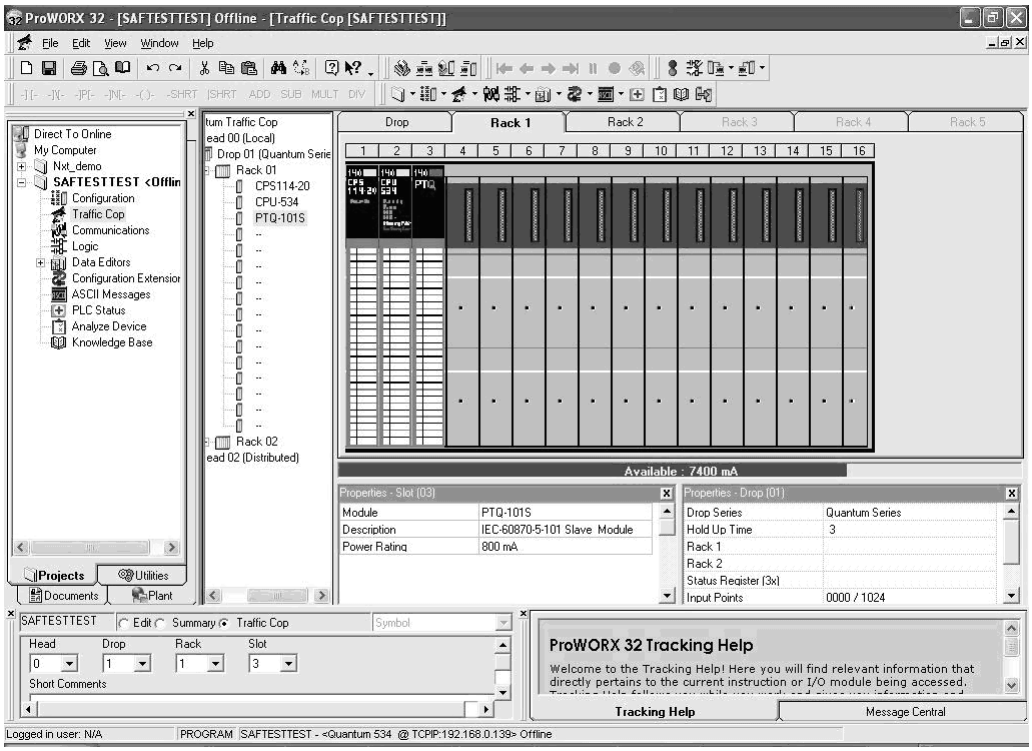
3 Select the .SaF File that is located at the CD-ROM shipped with the PTQ module.



4 After you click on Open you should see the PTQ modules imported (select I/O series as Quantum):



Now you can close the Schneider alliances application and run the Proworx 32 software. At the Traffic Cop section, select the PTQ module to be inserted at the slot:



4 Configuring the Processor with UnityPro XL

In This Chapter

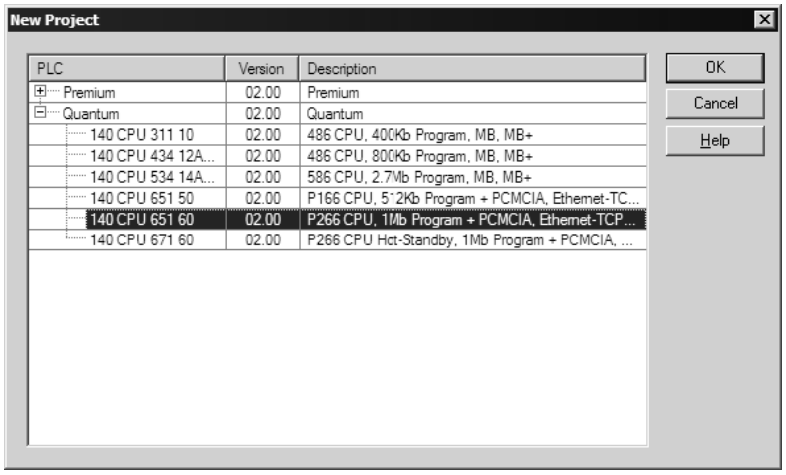
- Create a New Project 37
- Add the PTQ Module to the Project 39
- Build the Project 41
- Connect Your PC to the Processor 42
- Download the Project to the Processor 44

The following steps are designed to ensure that the processor (Quantum or Unity) is able to transfer data successfully with the PTQ module. As part of this procedure, you will use UnityPro XL to create a project, add the PTQ module to the project, set up data memory for the project, and then download the project to the processor.

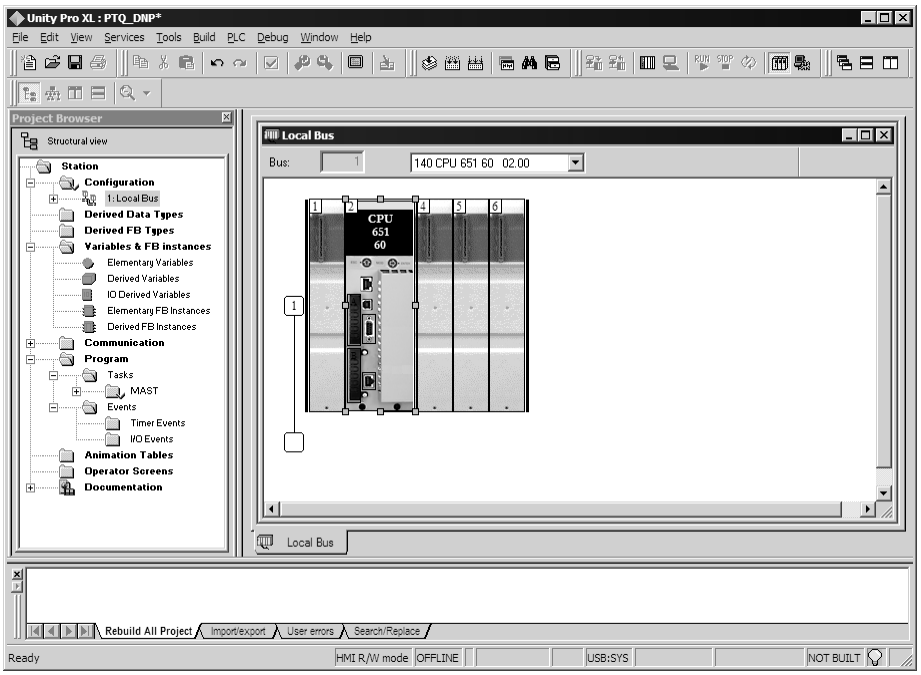
4.1 Create a New Project

The first step is to open UnityPro XL and create a new project.

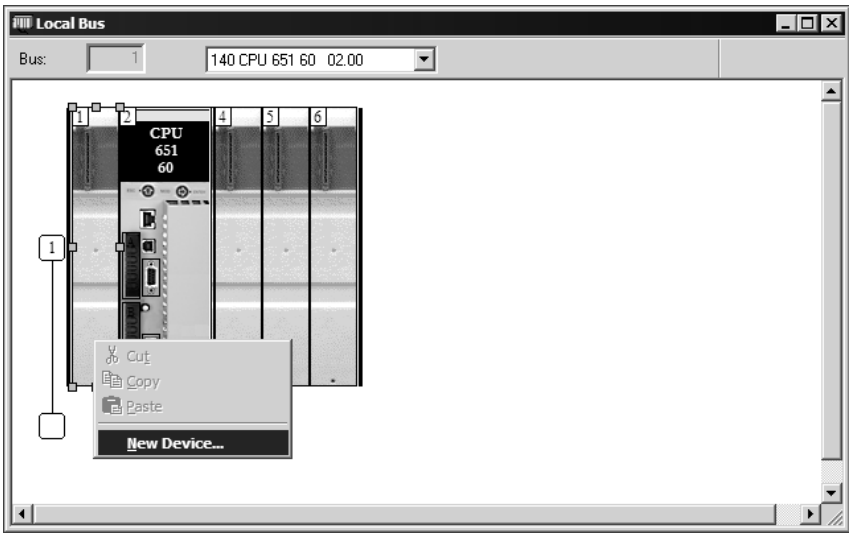
1 In the New Project dialog box, choose the CPU type. In the following illustration, the CPU is 140 CPU 651 60. Choose the processor type that matches your own hardware configuration, if it differs from the example. Click OK to continue.



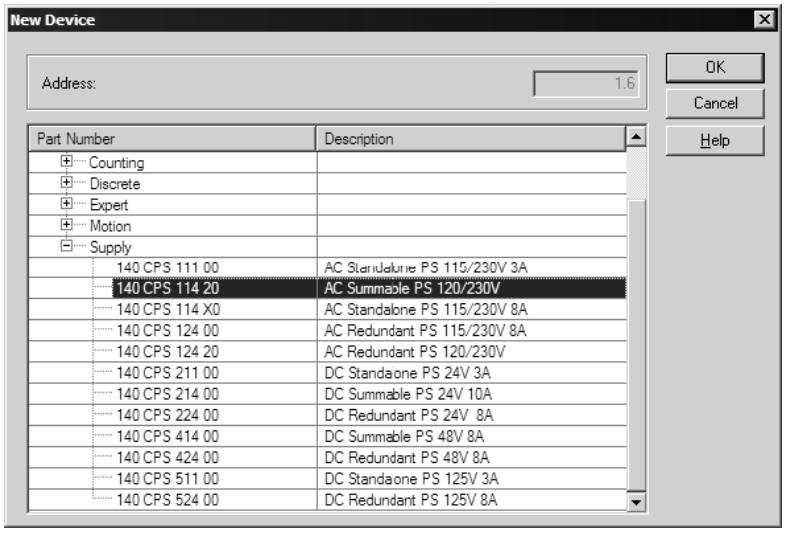
2 The next step is to add a power supply to the project. In the Project Browser, expand the Configuration folder, and then double-click the 1:LocalBus icon. This action opens a graphical window showing the arrangement of devices in your Quantum rack.



3 Select the rack position for the power supply, and then click the right mouse button to open a shortcut menu. On the shortcut menu, choose New Device..



4 Expand the Supply folder, and then select your power supply from the list. Click OK to continue.

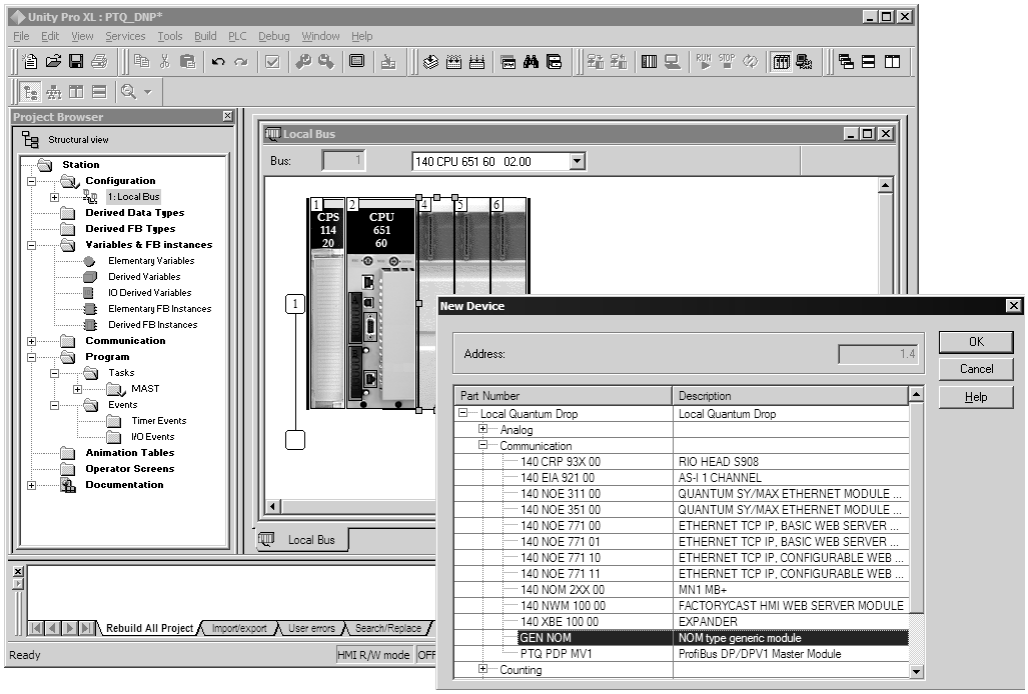


5 Repeat these steps to add any additional devices to your Quantum Rack.

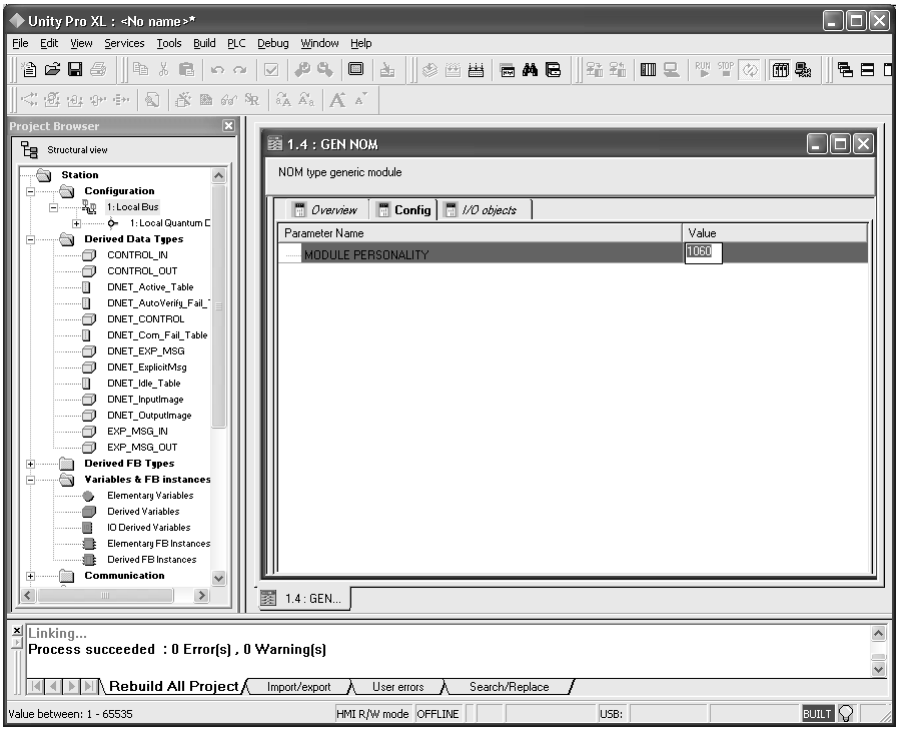
4.2 Add the PTQ Module to the Project

The next step is to add the PTQ module.

1 Expand the Communication tree, and select GEN NOM. This module type provides extended communication capabilities for the Quantum system, and allows communication between the PLC and the PTQ module without requiring additional programming.



2 Next, enter the module personality value. The correct value for ProTalk modules is 1060 decimal (0424 hex).



3 Before you can save the project in UnityProXL, you must validate the modifications. Open the Edit menu, and then choose Validate. If no errors are reported, you can save the project.

4 Save the project.

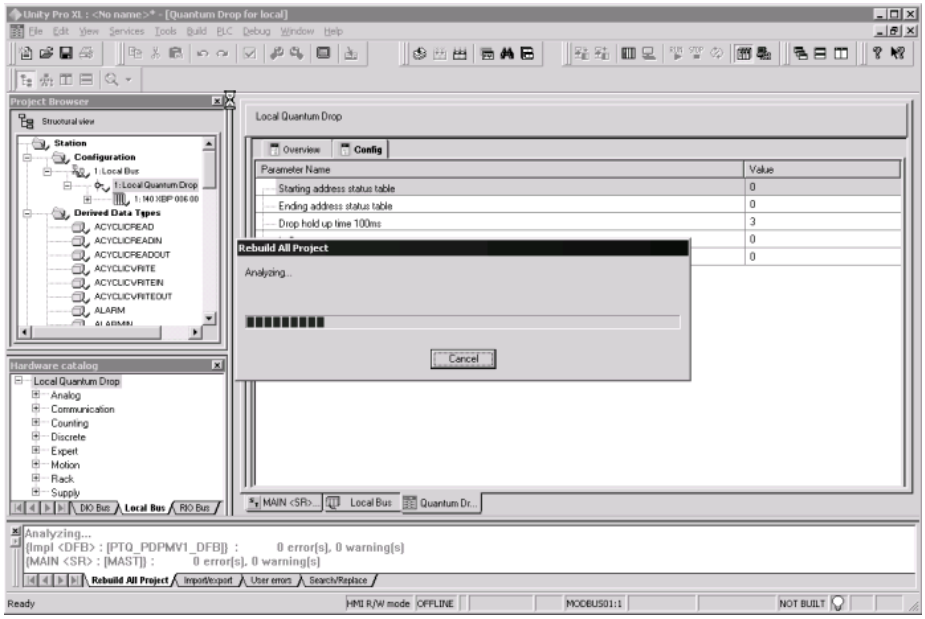
4.3 Build the Project

Whenever you update the configuration of your PTQ module or the processor, you must import the changed configuration from the module, and then build (compile) the project before downloading it to the processor.

Note: The following steps show you how to build the project in Unity Pro XL. This is not intended to provide detailed information on using Unity Pro XL, or debugging your programs. Refer to the documentation for your processor and for Unity Pro XL for specialized information.

To build (compile) the project:

1. Review the elements of the project in the Project Browser.
2. When you are satisfied that you are ready to download the project, open the Build menu, and then choose Rebuild all Project. This action builds (compiles) the project into a form that the processor can use to execute the instructions in the project file. This task may take several minutes, depending on the complexity of the project and the resources available on your PC.
3. As the project is built, Unity Pro XL reports its process in a Progress dialog box, with details appearing in a pane at the bottom of the window. The following illustration shows the build process under way.



After the build process is completed successfully, the next step is to download the compiled project to the processor.

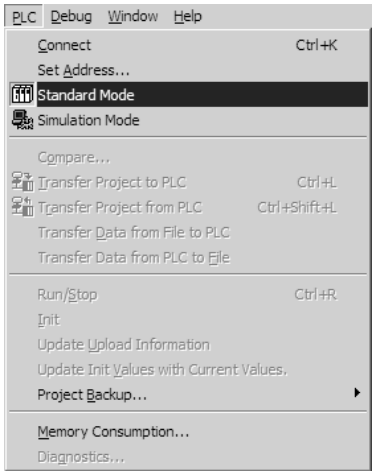
4.4 Connect Your PC to the Processor

The next step is to connect to the processor so that you can download the project file. The processor uses this project file to communicate over the backplane to modules identified in the project file.

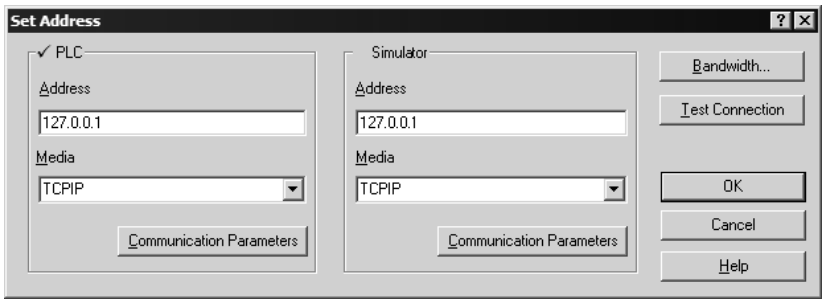
Note: If you have never connected from the PC to your processor before, you must verify that the necessary port drivers are installed and available to UnityPro XL.

To verify address and driver settings in UnityPro XL:

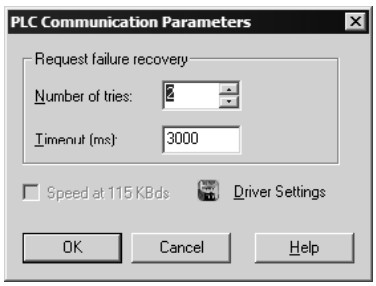
1. Open the PLC menu, and choose Standard Mode. This action turns off the PLC Simulator, and allows you to communicate directly with the Quantum or Unity hardware.



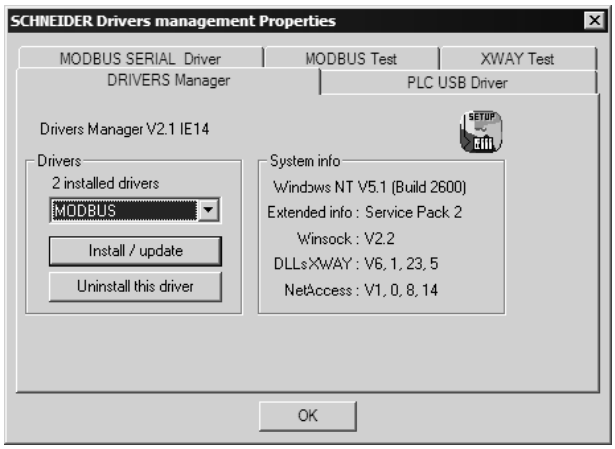
2. Open the PLC menu, and choose Set address... This action opens the Set address dialog box. Open the Media dropdown list and choose the connection type to use (TCPIP or USB).



3 If the Media dropdown list does not contain the connection method you wish to use, click the Communication Parameters button in the PLC area of the dialog box. This action opens the PLC Communication Parameters dialog box.



4 Click the Driver Settings button to open the SCHNEIDER Drivers management Properties dialog box.



5 Click the Install/update button to specify the location of the Setup.exe file containing the drivers to use. You will need your UnityPro XL installation disks for this step.

6 Click the Browse button to locate the Setup.exe file to execute, and then execute the setup program. After the installation, restart your PC if you are prompted to do so. Refer to your Schneider Electric documentation for more information on installing drivers for UnityPro XL.

4.4.1 Connecting to the Processor with TCPIP

The next step is to download (copy) the project file to the processor. The following steps demonstrate how to use an Ethernet cable connected from the Processor to your PC through an Ethernet hub or switch. Other connection methods may also be available, depending on the hardware configuration of your processor, and the communication drivers installed in UnityPro XL.

1. If you have not already done so, connect your PC and the processor to an Ethernet hub.
2. Open the PLC menu, and then choose Set address.

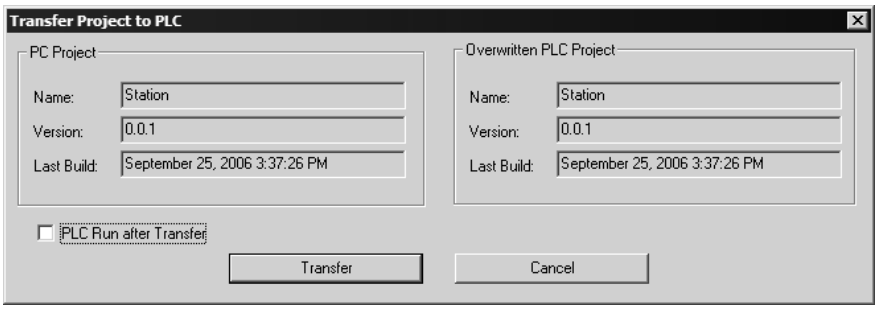
- Important: Notice that the Set address dialog box is divided into two areas. Enter the address and media type in the PLC area of the dialog box, not the Simulator area.

3. Enter the IP address in the address field. In the Media dropdown list, choose TCPIP.
4. Click the Test Connection button to verify that your settings are correct.

The next step is to download the Project to the Processor.

4.5 Download the Project to the Processor

1. Open the PLC menu and then choose Connect. This action opens a connection between the Unity Pro XL software and the processor, using the address and media type settings you configured in the previous step.
2. On the PLC menu, choose Transfer Project to PLC. This action opens the Transfer Project to PLC dialog box. If you would like the PLC to go to "Run" mode immediately after the transfer is complete, select (check) the PLC Run after Transfer check box.



3 Click the Transfer button to download the project to the processor. As the project is transferred, Unity Pro XL reports its process in a Progress dialog box, with details appearing in a pane at the bottom of the window.

When the transfer is complete, place the processor in Run mode.

5 Setting Up the ProTalk Module

In This Chapter

- Install the ProTalk Module in the Quantum Rack 47
- Connect the PC to the ProTalk Configuration/Debug Port 53

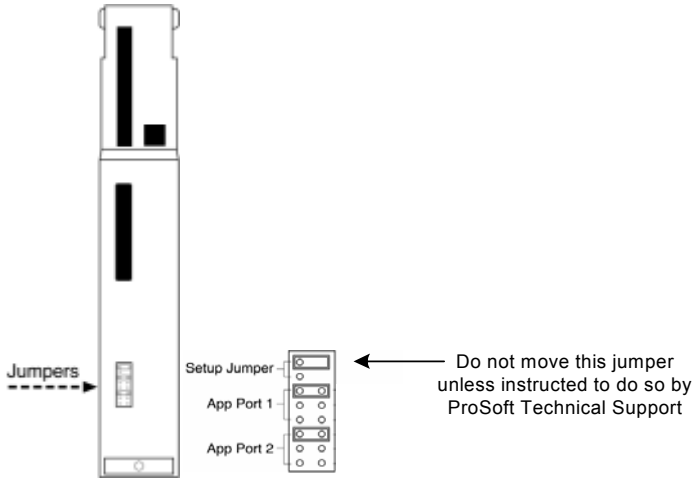
After you complete the following procedures, the ProTalk module will actively be transferring data bi-directionally with the processor.

5.1 Install the ProTalk Module in the Quantum Rack

5.1.1 Verify Jumper Settings

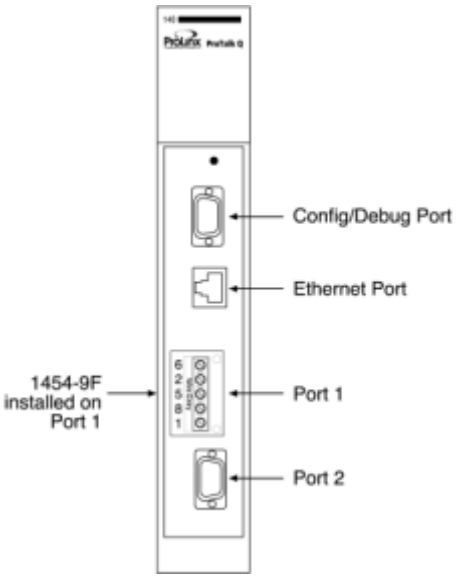
ProTalk modules are configured for RS-232 serial communications by default. To use RS-422 or RS-485, you must change the jumpers.

The jumpers are located on the back of the module as shown in the following illustration:


Jumpers
Setup Jumper
App Port 1
App Port 2
Do not move this jumper unless instructed to do so by ProSoft Technical Support

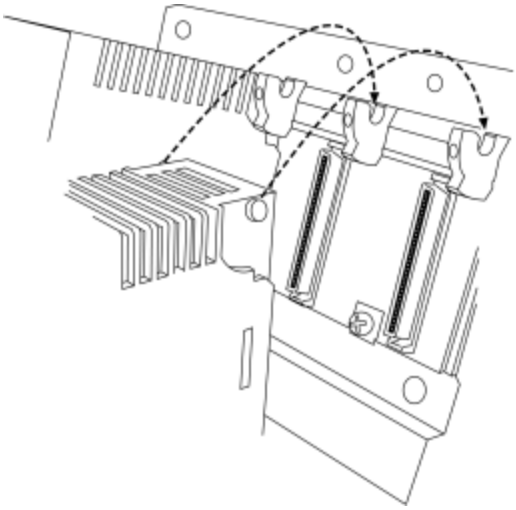
5.1.2 Inserting the 1454-9F connector

Insert the 1454-9F connector as shown. Wiring locations are shown in the table:

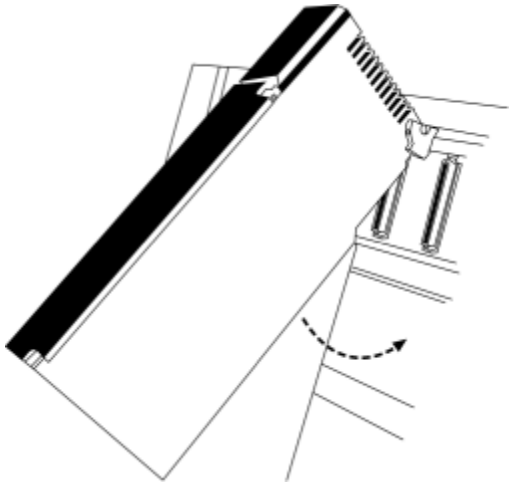


5.1.3 Install the ProTalk Module in the Quantum Rack

1. Place the Module in the Quantum Rack. The ProTalk module must be placed in the same rack as the processor.
2. Tilt the module at a 45° angle and align the pegs at the top of the module with slots on the backplane.



3 Push the module into place until it seats firmly in the backplane.



CaUTION: The PTQ module is hot-swappable, meaning that you can install and remove it while the rack is powered up. You should not assume that this is the case for all types of modules unless the user manual for the product explicitly states that the module is hot-swappable. Failure to observe this precaution could result in damage to the module and any equipment connected to it.

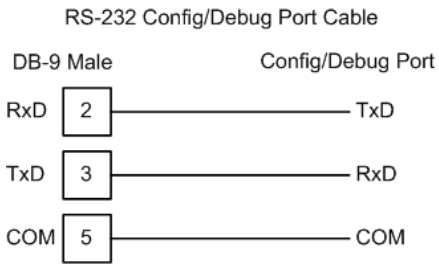
5.1.4 Cable Connections

The application ports on the PTQ-MCM module support RS-232, RS-422, and RS-485 interfaces. Please inspect the module to ensure that the jumpers are set correctly to correspond with the type of interface you are using.

Note: When using RS-232 with radio modem applications, some radios or modems require hardware handshaking (control and monitoring of modem signal lines). Enable this in the configuration of the module by setting the UseCTS parameter to 1.

RS-232 Configuration/Debug Port

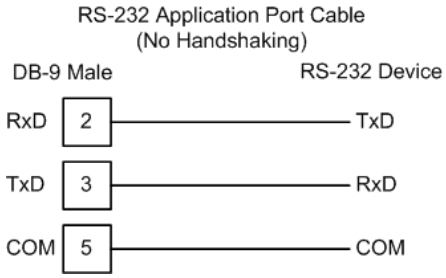
This port is physically a DB-9 connection. This port permits a PC based terminal emulation program to view configuration and status data in the module and to control the module. The cable for communications on this port is shown in the following diagram:



The Ethernet port on this module (if present) is inactive.

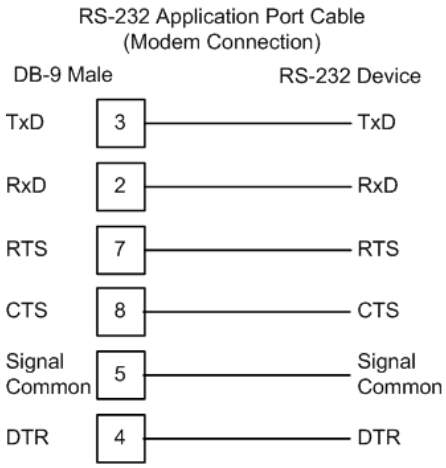
RS-232

When the RS-232 interface is selected, the use of hardware handshaking (control and monitoring of modem signal lines) is user definable. If no hardware handshaking will be used, the cable to connect to the port is as shown below:



RS-232: Modem Connection

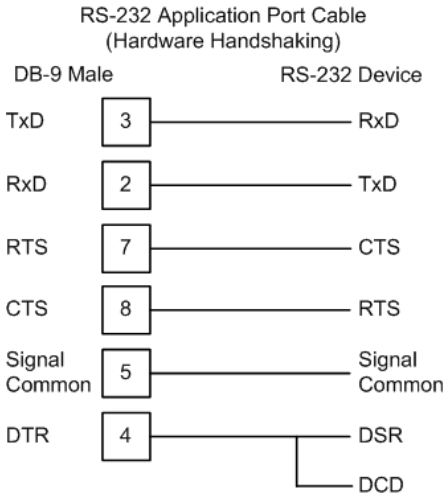
This type of connection is required between the module and a modem or other communication device.



The "Use CTS Line" parameter for the port configuration should be set to 'Y' for most modem applications.

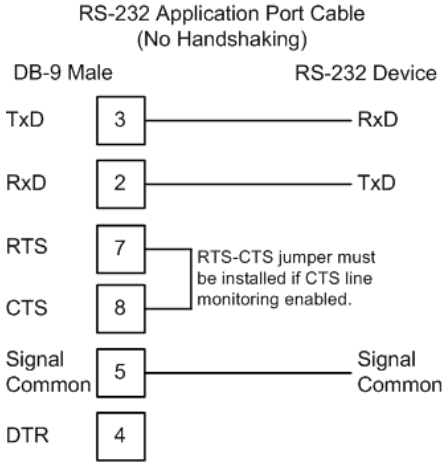
RS-232: Null Modem Connection (Hardware Handshaking)

This type of connection is used when the device connected to the module requires hardware handshaking (control and monitoring of modem signal lines).



RS-232: Null Modem Connection (No Hardware Handshaking)

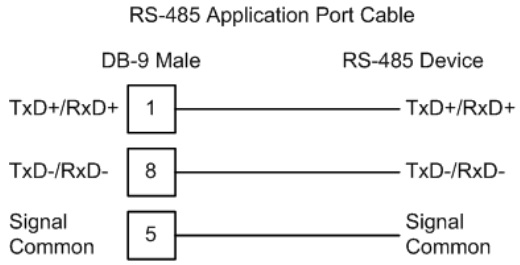
This type of connection can be used to connect the module to a computer or field device communication port.



Note: If the port is configured with the "Use CTS Line" set to 'Y', then a jumper is required between the RTS and the CTS line on the module connection.

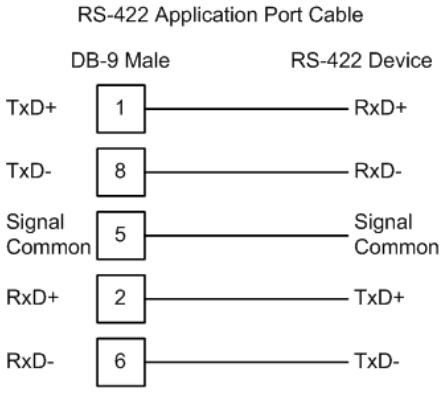
RS-485

The RS-485 interface requires a single two or three wire cable. The Common connection is optional and dependent on the RS-485 network. The cable required for this interface is shown below:



Note: Terminating resistors are generally not required on the RS-485 network, unless you are experiencing communication problems that can be attributed to signal echoes or reflections. In this case, install a 120 ohm terminating resistor on the RS-485 line.

RS-422



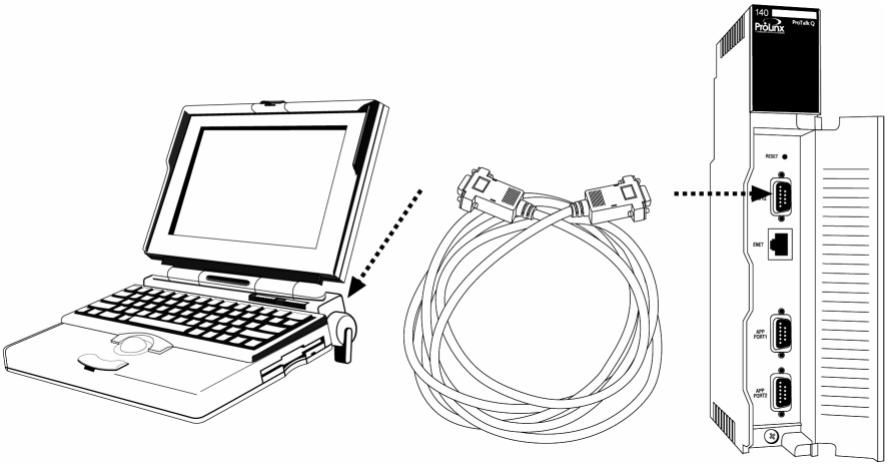
RS-485 and RS-422 Tip

If communication in the RS-422/RS-485 mode does not work at first, despite all attempts, try switching termination polarities. Some manufacturers interpret +/- and A/B polarities differently.

5.2 Connect the PC to the ProTalk Configuration/Debug Port

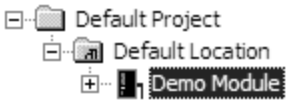
Make sure you have exited the Quantum programming software before performing these steps. This action will avoid serial port conflict.

Using the supplied Null Modem cable, connect your PC or Laptop to the Configuration/Debug port on the ProTalk module as shown

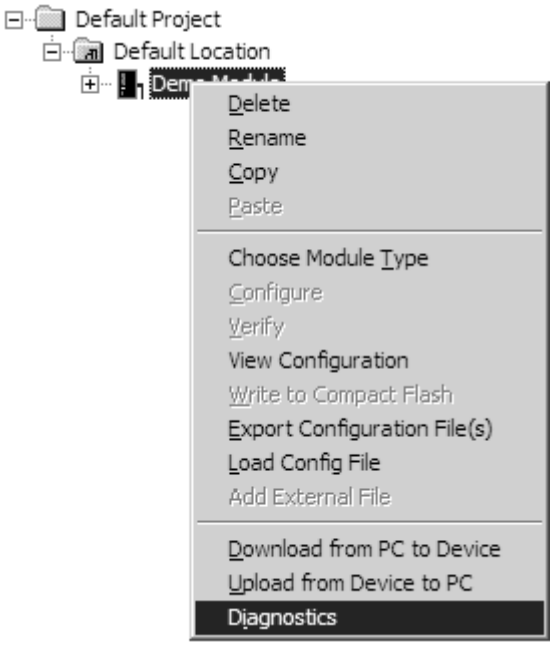


To connect to the module's Configuration/Debug serial port:

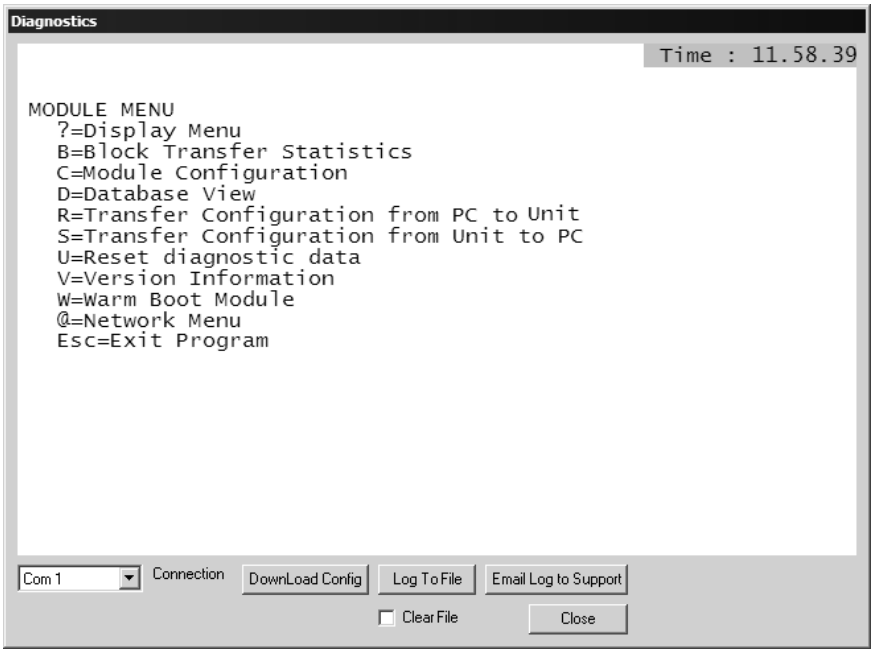
1 Start PCB program with the application file to be tested. Right click over the module icon.



2 On the shortcut menu, choose Diagnostics.



3 This action opens the Diagnostics dialog box. Press "?" to display the Main Menu.



Important: The illustrations of configuration/debug menus in this section are intended as a general guide, and may not exactly match the configuration/debug menus in your own module.

If there is no response from the module, follow these steps:

1. Verify that the null modem cable is connected properly between your computer's serial port and the module. A regular serial cable will not work.
2. On computers with more than one serial port, verify that your communication program is connected to the same port that is connected to the module.
3. If you are still not able to establish a connection, contact ProSoft Technology for assistance.

6 Modifying the Configuration File

In This Chapter

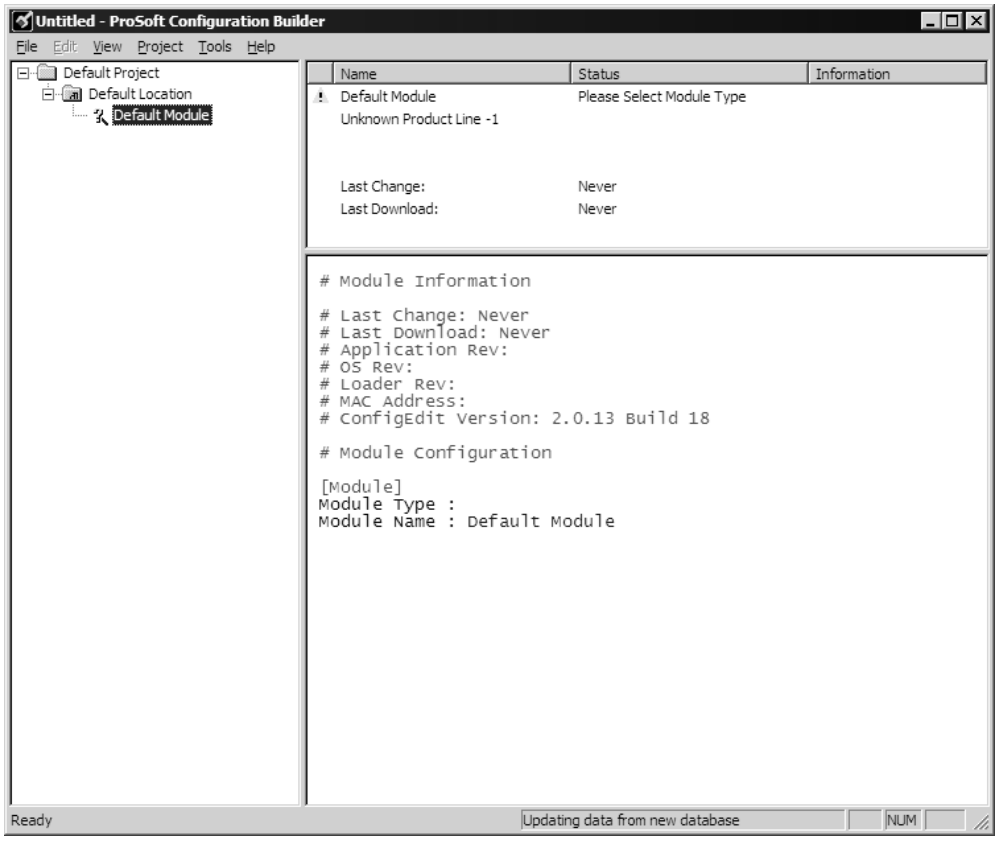
- ProSoft Configuration Builder 57
- Edit the Configuration File 62
- Download the Project to the Module 76
- Verification and Troubleshooting 77

6.1 ProSoft Configuration Builder

ProSoft Configuration Builder (PCB) provides a quick and easy way to manage module configuration files customized to meet your application needs. PCB is not only a powerful solution for new configuration files, but also allows you to import information from previously installed (known working) configurations to new projects.

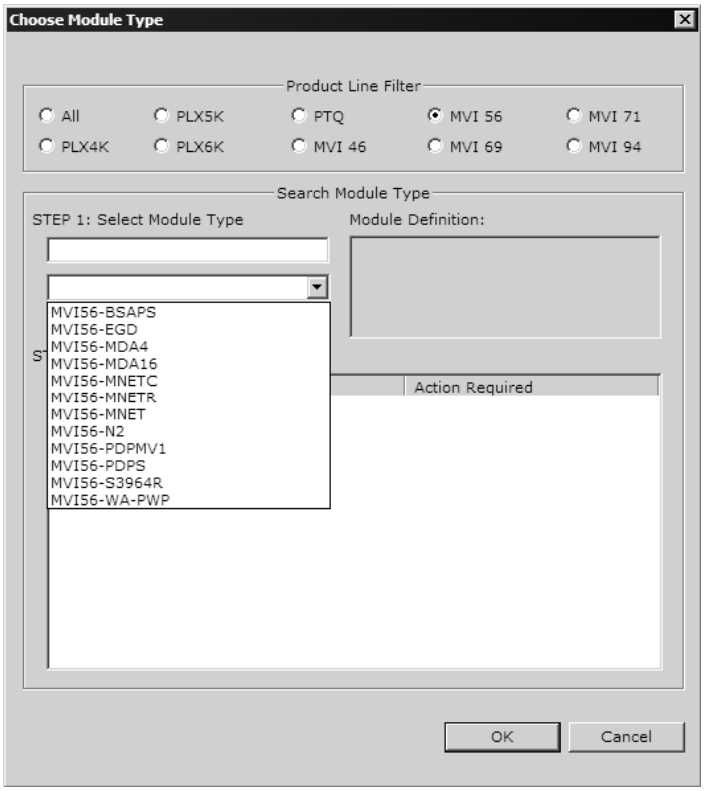
6.1.1 Set Up the Project

To begin, start ProSoft Configuration Builder. If you have used other Windows configuration tools before, you will find the screen layout familiar. ProSoft Configuration Builder's window consists of a tree view on the left, an information pane and a configuration pane on the right side of the window. When you first start ProSoft Configuration Builder, the tree view consists of folders for Default Project and Default Location, with a Default Module in the Default Location folder. The following illustration shows the ProSoft Configuration Builder window with a new project.



Your first task is to add the PTQ-MCM module to the project.

1. Use the mouse to select "Default Module" in the tree view, and then click the right mouse button to open a shortcut menu.
2. On the shortcut menu, choose "Choose Module Type". This action opens the Choose Module Type dialog box.



3. In the Product Line Filter area of the dialog box, select PTQ. In the Select Module Type dropdown list, select PTQ-MCM, and then click OK to save your settings and return to the ProSoft Configuration Builder window.

The next task is to set the module parameters.

Adding a Project

To add a project to an existing project file:

1. Select the Default Project icon.
2. Choose Project from the Project menu, then choose Add Project. A new project folder appears.

Adding a Module

To add a module to your project:

1 Double-click the Default Module icon to open the Choose Module Type dialog box.
2 On the Choose Module Type dialog box, select the module type.

or

1 Open the Project menu and choose Location.
2 On the Location menu, choose Add Module.

To add a module to a different location:

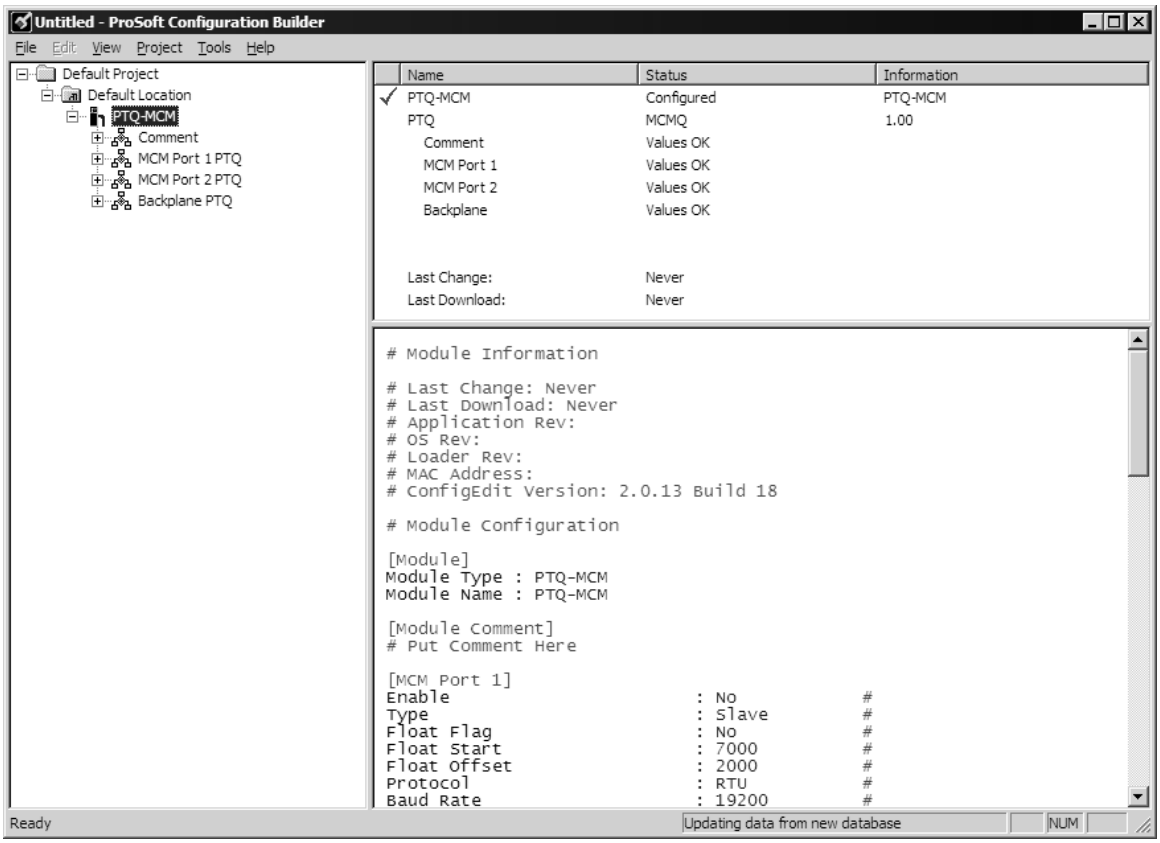
1 Right-click the Location folder and choose Add Module. A new module icon appears.

or

1 Select the Location icon.
2 From the Project menu, select Location, then select Add Module.

6.1.2 Set Module Parameters

Notice that the contents of the information pane and the configuration pane changed when you added the PTQ-MCM module to the project.



At this time, you may wish to rename the "Default Project" and "Default Location" folders in the tree view.

To rename an object:

1. Select the object, and then click the right mouse button to open a shortcut menu. From the shortcut menu, choose Rename.
2. Type the name to assign to the object.
3. Click away from the object to save the new name.

Module Entries

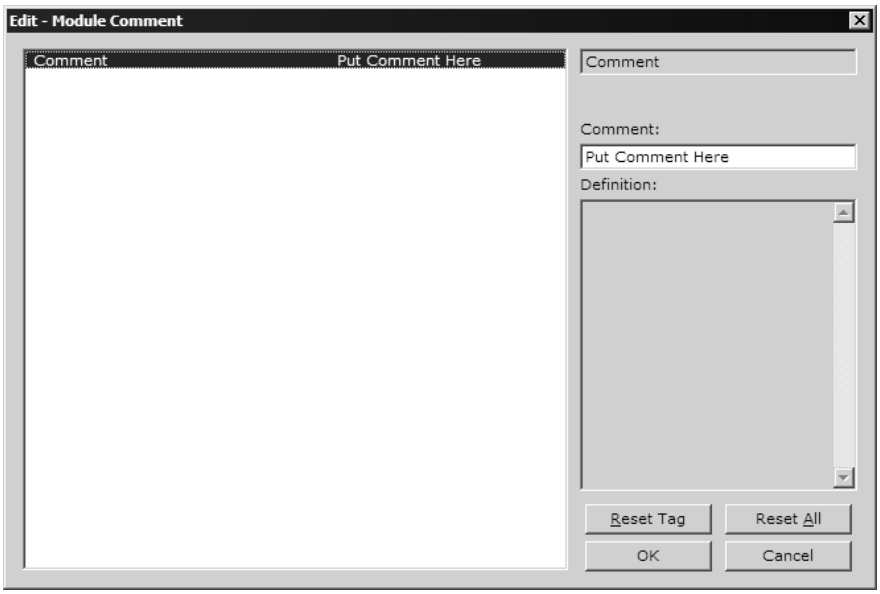
To configure module parameters

1. Click on the plus sign next to the icon Comment to expand module information.
2. Double-click the Module Comment icon to open the Edit dialog box.
3. To edit a parameter, select the parameter in the left pane and make your changes in the right pane.
4. Click OK to save your changes.

Comment Entries

To add comments to your configuration file:

1. Click the plus sign to the left of the Comment icon to expand the Module Comments.
2. Double-click the Module Comment icon. The Edit - Module Comment dialog appears.



3. Enter your comment and click OK to save your changes.

Printing a Configuration File

To print a configuration file:

1. Select the Module icon, and then click the right mouse button to open a shortcut menu.
2. On the shortcut menu, choose View Configuration. This action opens the View Configuration window.
3. On the View Configuration window, open the File menu, and choose Print. This action opens the Print dialog box.
4. On the Print dialog box, choose the printer to use from the dropdown list, select printing options, and then click OK.

6.2 Edit the Configuration File

Note: It is important that you plan your configuration before modifying the configuration files. The remainder of this step provides the information to make the appropriate modifications to the configuration files.
Important: This module supports a maximum configuration file size of 128 kilobytes (131072 bytes). If the configuration file is larger than this size, the module will not accept the download. You can reduce the size of the configuration file by opening the file in a text editor and removing comment lines (lines preceded with the # character).

The PTQMCM.CFG file has the following main sections:

- [Module]
- [Backplane Configuration]
- [MCM Port 1]
- [MCM Port 1 Commands]
- [MCM Port 2]
- [MCM Port 2 Commands]

Important notes to consider when editing the sample configuration file:

- Comments within the file are preceded by the pound (#) sign. Any text on a line that occurs after the # character will be ignored.
- Do not use tabs or other non-printing characters instead of spaces to separate parameters (spacebar).
- Parameter names must begin in the first column of a line, and may not be preceded with a space (spacebar) or other non-printing character.

6.2.1 [Module]

```
# This section is used to define the configuration for the Module level
# data.
#
[Module]
Module Type         : PTQ-MCM
Module Name         : Test Example of PTQ-MCM Communication Module
```

Module Type Parameter

```
Module Type         : PTQ-MCM
```

The Module Type parameter is used to assign a name to the module that can be viewed using the configuration/debug port. Use this parameter to identify the module and the configuration file. You can enter a name from 0 to 80 characters.

Module Name

0 to 80 characters

This parameter assigns a name to the module that can be viewed using the configuration/debug port. Use this parameter to identify the module and the configuration file.

6.2.2 [Backplane Configuration]

```
[Backplane Configuration]
#These values are required to define the data area to transfer between the
module
#and the processor.
Read Register Start :     0  #Database start register to move to processor
Read Register Count :   100  #Number of words moved from module to processor
Write Register Start:   500  #Database start register where data placed from
                             #processor
Write Register Count:   100  #Number of words moved from processor to module

#Used to define the area in the Processor for the module to interface with
3x Register Start:        1  #3x start register where data moved from module to
                             #processor (1-n)
4x Register Start:        1  #4x start register where data moved from processor
                             #to module (1-n)

Initialize Output Data       : No   #Initialize the database file on startup
Failure Flag Count           : 0    #
Error/Status Block Pointer   : 3000 #Number of register to store error and
                                    #status
```

Read Register Start

Range 0 to 3999

This parameter specifies the starting register in the module where the data transferred from the processor will be placed. Valid range for this parameter is 0 to 3999.

Read Register Count

Range 0 to 3999

This parameter specifies the number of registers to be transferred from the module to the processor. Valid entry for this parameter is 0 to 3999.

Write Register Start

Range 0 to 3999

This parameter specifies the starting register in the module where data will be transferred from the module to the processor. Valid range for this parameter is 0 to 3999.

Write Register Count

Range 0 to 3999

This parameter specifies the number of registers to be transferred from the module to the processor. Valid entry for this parameter is 0 to 3999

3x Register Start

1 to n

This parameter sets the first register in the processor where the data transferred from the module to the processor will be placed.

4x Register Start

1 to n

This parameter sets the first register in the processor where the data transferred from the processor to the module is present.

Initialize Output Data

Yes or No

This parameter determines if the output data for the module should be initialized with values from the processor. If the value is set to 0, the output data will be initialized to 0. If the value is set to 1, the data will be initialized with data from the processor. Use of this option requires associated ladder logic to pass the data from the processor to the module.

Failure Flag Count

0 through 65535

This parameter specifies the number of successive transfer errors that must occur before the communication ports are shut down. If the parameter is set to 0, the communication ports will continue to operate under all conditions. If the value is set larger than 0 (1 to 65535), communications will cease if the specified number of failures occur.

Error/Status Block Pointer

1 to 7000

The Error/Status Block Pointer parameter is used to specify the range of database registers to use for error and status data. The value should be no lower than 700, to avoid overwriting data, and no higher than 6970, to allow sufficient space for the error/status block.

6.2.3 [MCM Port X]

The information in this section applies to both Port 1 and Port 2.

```
# This section is used to define the port 1 configuration for the MCM device
#
[MCM Port 1]
Enable                 :   Yes  #No=Port Disabled,Yes=Port Enabled
Type                   :     5  #Type of port (0=Master, 1=Slave, 2=Block PT,
                                  #3=Block PT Swap, 4=Direct PT, 5=Direct PT
Swap)
Pass Thru Address      :   300  #Address where pass-thru block id is used only
                                  #Type equal to 2 or 3
Float Flag             :    No  #Float Flag (No=No Floating Point Data,
                                  #Yes=Use Floating Point Data)
Float Start            :  4000  #The first register of floating-point data.
Float Offset           :  2000  #Internal DB offset to start of floating point
                                  #data
Protocol               :     r  #ASCII or RTU
Baud Rate              :   192  #Baud rate for port (300, 600, 1200, 2400,
4800,
                                  #9600, 19200, 38400, 57600, 115(for 115200))
Parity                 :  None  #N=None,O=Odd,E=Even
Data Bits              :     8  #7 or 8 data bits for messages
Stop Bits              :     1  #1 or 2 stop bits for messages
RTS On                 :     0  #Delay after RTS set before message sent (mSec)
RTS Off                :     0  #Delay after message before RTS dropped (mSec)
Minimum Response Delay :     0  #Number of mSec to delay before response
Use CTS Line           :    No  #Monitor CTS modem line (Y/N)
Slave Address          :     1  #0-255 Slave Node Address
Bit Input Offset       :     0  #Internal DB offset to bit input data (Slave)
Word Input Offset      :     0  #Internal DB offset to word input data (Slave)
Output Offset          :     0  #Internal DB offset to bit output data (Slave)
Hold Offset            :     0  #Internal DB offset to bit output data (Slave)
Command Count          :     3  #Command list count (Master)
Min Cmd Delay          :     1  #Minimum time between each command
                                  #(Master 0-65535)
Cmd Err Pointer        :  4500  #Internal DB location to place command error
                                  #list (Master 0-4900)
Response Timeout       :     0  #Response timeout for command (Master)
Retry Count            :     2  #Retry count for failed request (Master)
Error Delay Count      :     0  #Command cycle count if error (Master)
```

Enable

```
Enable           :   Yes  #No=Port Disabled,Yes=Port Enabled
```

This parameter is used to define if this Modbus port will be used. If the parameter is set to 0, the port is disabled. A value of 1 enables the port.

Type

```
Type                    :     0  #Type of port (0=Master, 1=Slave,
                                  #2=Block Pass-Thru, 3=Block PT w/ Byte Swap,
                                  #4=Direct PT, 5=Direct PT w/ Byte Swap)
```

This parameter specifies which device type the port will emulate. For specific information on types 2-5, refer to the Pass-Thru Functionality section of this manual.

Pass Thru Address

```
Pass Thru Address       :  1000  #Pass-Thru address to be used during Block
                                  #Pass-Thru mode.
```

This parameter specifies the starting processor holding register address where the pass-thru block will be copied to. This parameter is only used if the type parameter is configured for Block-Pass-Thru operation (codes 2 and 3). The module adds 400001 to the actual value configured by the module. For example, a value of 200 would imply in a processor address of 400201

Float Flag

```
Float Flag         :    No  #Float Flag (No=No Floating Point Data, Yes=Use
```

This flag specifies if the floating-point data access functionality is to be implemented. If the float flag is set to 1, Modbus functions 3, 6, and 16 will interpret floating-point values for registers as specified by the two following parameters.

Float Start

```
Float Start        :  7000  #The first register of floating-point data.
```

This parameter defines the first register of floating-point data. All requests with register values greater than or equal to this value will be considered floating-point data requests. This parameter is only used of the Float Flag is enabled.

Float Offset

```
Float Offset       :  2000  #Internal DB offset to start of floating point
                             #data
```

This parameter defines the start register for floating-point data in the internal database. This parameter is only used if the Float Flag is enabled.

Protocol

ASCII or RTU

This parameter specifies the Modbus protocol to be used on the port. Valid protocols are: rtu = Modbus RTU and ascii = Modbus ASCII.

Baud Rate

```
Baud Rate         :  4800  #Baud rate for port (300, 600, 1200, 2400, 4800,
                            #9600, 19200, 38400, 57600, 115(for 115200))
```

This is the baud rate to be used on the port. Enter the baud rate as a value. For example, to select 19K baud, enter 19200. Valid entries are 110, 150, 300, 600, 1200, 2400, 4800, 9600, 19200, 28800, 38400, 576, and 115.

Parity

```
Parity            :  None  #N=None,O=Odd,E=Even
```

This is the parity code to be used for the port. Values are None, Odd, Even.

Data Bits

```
Data Bits         :     8  #7 or 8 data bits for messages
```

This parameter sets the number of data bits for each word used by the protocol. Valid entries for this field are 7 or 8.

Stop Bits

```
Stop Bits         :     1  #1 or 2 stop bits for messages
```

This parameter sets the number of stop bits to be used with each data value sent. Valid entries are 1 and 2.

RTS On

```
RTS On            :     0  #Delay after RTS set before message sent (mSec)
```

This parameter sets the number of milliseconds to delay after RTS is asserted before the data will be transmitted. Valid values are in the range of 0 to 10000 milliseconds.

RTS Off

```
RTS Off           :     0  #Delay after message before RTS dropped (mSec)
```

This parameter sets the number of milliseconds to delay after the last byte of data is sent before the RTS modem signal will be set low. Valid values are in the range of 0 to 10000 milliseconds.

Minimum Response Delay

```
Minimum Response Delay :     0  #Number of mSec to delay before response
```

This parameter sets the number of milliseconds to delay after the last byte of data is sent before the RTS modem signal will be set low. Valid values are in the range of 0 to 10000 milliseconds. When the port is configured for Modbus Slave operation, increase this parameter to allow the slave to send the message response to the port for the master.

Use CTS Line

```
Use CTS Line      :    No  #Monitor CTS modem line (Y/N)
```

This parameter specifies if the CTS modem control line is to be used. If the parameter is set to 0, the CTS line will not be monitored. If the parameter is set to 1, the CTS line will be monitored and must be high before the module will send data. This parameter is normally only required when half-duplex modems are used for communication (2-wire).

Slave Address

```
Slave Address     :     0  #0-255 Slave Node Address
```

This parameter defines the virtual Modbus slave address for the internal database. All requests received by the port with this address are processed by the module. Verify that each device has a unique address on a network. Valid range for this parameter is 1 to 255 (247 on some networks). The address 0 is used for broadcast operations. When the slave port is configured without address 0 it will process all write commands received from the master but it will not send a message response.

Bit Input Offset

```
Bit Input Offset  :     0  #Internal DB offset to bit input data (Slave)
```

This parameter specifies the offset address in the internal Modbus database that is to be used with network requests for Modbus Function 2 commands. For example, if the value is set to 150, an address request of 0 will return the value at register 150 in the database.

Word Input Offset

```
Word Input Offset :     0  #Internal DB offset to word input data (Slave)
```

This parameter specifies the offset address in the internal Modbus database that is to be used with network requests for Modbus Function 4 commands. For example, if the value is set to 150, an address request of 0 will return the value at register word 150 in the database.

Output Offset

```
Output Offset     :     0  #Internal DB offset to bit output data (Slave)
```

This parameter specifies the offset address in the internal Modbus database that is to be used with network requests for Modbus function 1, 5, or 15 commands. For example, if the value is set to 50, an address request of 0 will correspond to register word 50 (bit 800) in the database.

Hold Offset

0 to 4999

This parameter specifies the offset address in the internal Modbus database that is to be used with network requests for Modbus function 3, 6, or 16 commands. For example, if a value of 50 is entered, a request for address 0 will correspond to the register 50 in the database.

Command Count

0 to 100

This parameter specifies the number of commands to be processed by the Modbus master port.

Min Cmd Delay

0 to 65535

This parameter specifies the number of milliseconds to wait between issuing each command. This delay value is not applied to retries.

Cmd Err Pointer

-1 to 3995

This parameter sets the address in the internal Modbus database where the command error will be placed. If the value is set to -1, the data will not be transferred to the database. The valid range of values for this parameter is -1 to 4999. For example, if this parameter is configured for 1000, the command errors will be copied to the database as follows:

1000: error code for command 0

1001: error code for command 1

etc.

An error code of 0 means that the command was successfully sent (no error). Refer to Error Status Table for the command error code listings.

Response Timeout

```
Response Timeout  :     0  #Response timeout for command (Master)
```

This parameter represents the message response timeout period in 1-millisecond increments. This is the time that a port configured as a master will wait before transmitting a command if no response is received from the addressed slave. The value is set depending upon the communication network used and the expected response time of the slowest device in the network.

Retry Count

```
Retry Count       :     0  #Retry count for failed request (Master)
```

This parameter specifies the number of times a command will be retried if it fails. If the master port does not receive a response after the last retry, the slave device's communication will be suspended on the port for Error Delay Counter scans.

Error Delay Counter

0 to 65535

This parameter specifies the number of polls to skip on the slave before trying to re-establish communications. After the slave fails to respond, the master will skip commands to be sent to the slave the number of times entered in this parameter.

6.2.4 [Modbus Port 1 Commands]

```
[Modbus Port 1 Commands]
#     Internal   Poll  Reg Swap   Node Modbus MB Address
# Enable  Address Interval Count Code Address  Func  in Device
START
    1     0     0   10   0   255   16     100 #
    1     0     0   10   1   255   16     110 #
    1     0     0   10   2   255   16     120 #
    1     0     0   10   3   255   16     130 #
END
```

You should configure the following parameters for each command:

Parameter	Description
Enable	This parameter defines how the command will be sent. The following codes are valid: 0: Disabled Disables the command. This command could still be activated from the processor using the Command Control special block (requires backplane command code 3) 1: Continuous The command will be issued with a frequency determined by the poll interval parameter that is used to define minimum delays between consecutive commands. 2: Conditional The conditional command is sent when the source data changes. It only applies to modbus write commands (modbus functions 5,6, 15 and 16)
Internal Address	Configure the PTQ-MCM database address associated with the modbus command. For a read command: this is the database address where the data read from the slave device will be copied to. For a write command: this is the database address where the data written to the slave device will be copied from. For bit commands (for example: 1, 2, 5 and 15) the module expects this value to represent the bit-address in the database. For example, a value of 100 would actually configure the Internal Address as BIT 100. For word commands (for example: 3, 4, 6 and 16) the module expects this value to represent the word-address in the database. For example, a value of 100 would actually configure the Internal Address as WORD 100 (1 word = 2 bytes).

Parameter	Description
Poll Interval	The minimum delay to be applied between consecutive polls for this command. Enter this number as milliseconds.
Reg Count	Register count. Enter the number of bits or words for this command. The maximum value for word commands (3, 4, 6 and 16) is 125 words. The maximum value for word commands (1, 2, 5 and 15) is 800 bits.
Swap Code	This value configures if the data bytes and words should be swapped. The possible values are:

Swap Code	Description
0	None - No Change is made in the byte ordering (1234 = 1234)
1	Words - The words are swapped (1234=3412)
2	Words & Bytes - The words are swapped then the bytes in each word are swapped (1234=4321)
3	Bytes - The bytes in each word are swapped (1234=2143)

Node Address	The slave address to which the command will be sent. This parameter can be assigned numbers from 2 to 255.
Modbus Function	The following modbus functions are supported: 1- Read Coil Status (bit) 2- Read Input Status (bit) 3- Read Holding Registers (word - 16 bits) 4- Read Input Registers (word - 16 bits) 5- Force Single Coil (bit) 6- Preset Single Register (word - 16 bits) 15- Force Multiple Coils (bit) 16- Preset Multiple Registers (word - 16 bits)
MB Address in Device	This parameter defines the starting address in the device being considered by the command. Values entered in this field are dependent on the node's database definition. Refer to the specific manufacturer's database definition for the device to determine the location of the data to be interfaced.

ENRON Floating Point Support

Many manufacturers have implemented special support in their drivers to support what is commonly called the Enron version of the MODBUS protocol. In this implementation, register addresses > 7000 are presumed to be floating point values. The significance to this is that the count field now becomes a 'number of values' field. In floating point format, each value represents two words.

Configuring the Floating Point Data Transfer

A common question when using the module as a Modbus Master is how floating point data is handled. This really depends on the slave device and how it addresses this application.

Just because your application is reading/writing floating point data, does not mean that you must configure the Float Flag, Float Start parameters within the module.

These parameters are only used to support what is typically referred to as Enron or Daniel Modbus, where one register address must have 32 bits, or one floating point value. Below is an example:

Example #1

Modbus Address	Data Type	Parameter
47101	32 bit REAL	TEMP Pump #1
47102	32 bit REAL	Pressure Pump #1
47103	32 bit REAL	TEMP Pump #2
47104	32 bit REAL	Pressure Pump #2

With the module configured as a master, you only need to enable these parameters to support a write to this type of addressing (Modbus FC 6 or 16).

If the slave device shows addressing as shown in Example #2, then you need not do anything with the Float Flag, Float Start parameters, as they use two Modbus addresses to represent one floating point value:

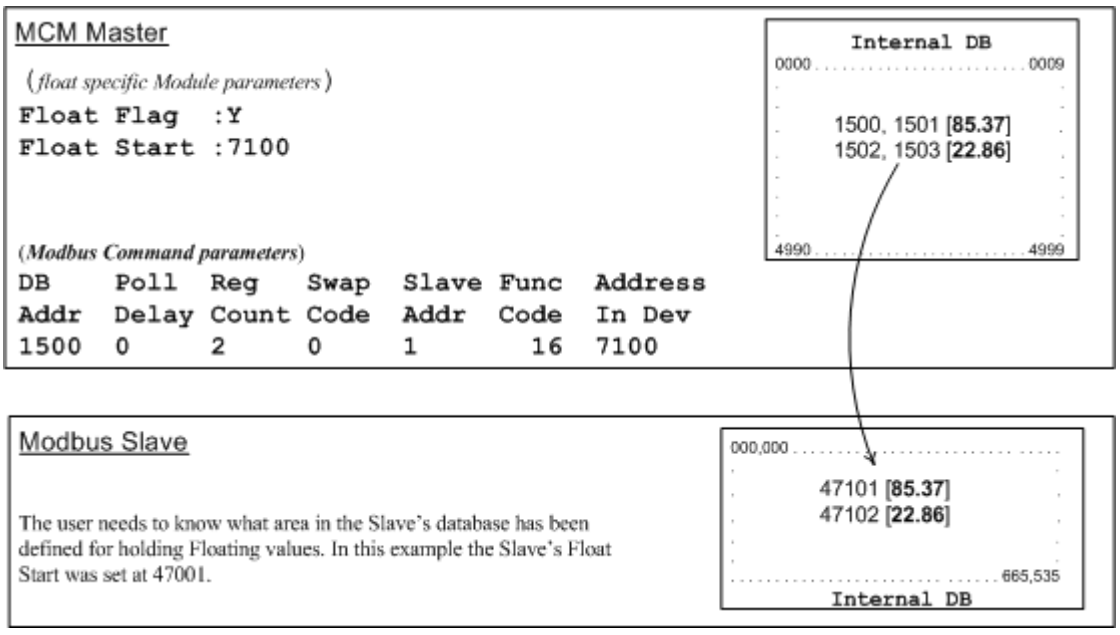
Example #2

Modbus Address	Data Type	Parameter
47101	32 bit REAL	TEMP Pump #1
47103	32 bit REAL	Pressure Pump #1
47105	32 bit REAL	TEMP Pump #2
47107	32 bit REAL	Pressure Pump #2

Because each 32 bit REAL value is represented by two Modbus Addresses (example 47101 and 47102 represent TEMP Pump #1), then you need not set the Float Flag, or Float Start for the module for Modbus FC 6 or 16 commands being written to the slave.

Below are specific examples:

Master is issuing Modbus command with FC 16 (with Float Flag: Yes) to transfer Float data to Slave.



(Float specific module parameters)

Float Flag: "Y" tells the Master to consider the data values that need to be sent to the Slave as floating point data where each data value is composed of 2 words (4 bytes or 32 bits).

Float Start: Tells the Master that if this address number is <= the address number in "Addr in Dev" parameter to double the byte count quantity to be included in the Command FC6 or FC16 to be issued to the Slave. Otherwise the Master will ignore the "Float Flag: Y" and treat data as composed of 1 word, 2 bytes.

(Modbus Command parameters)

DB Addr - Tells the Master where in its data memory is the beginning of data to obtain and write out to the Slave (slave) device.

Reg Count - Tells the Master how many data points to send to the Slave. Two counts will mean two floating points with Float Flag: Y and the "Addr in Dev" => the "Float Start" Parameter.

Swap Code - Tells the Master how to orient the Byte and Word structure of the data value. This is device dependent. Check Command Entry formats Section.

Func Code - Tells the Master to write the float values to the Slave. FC16.

Addr in Dev - Tells the Master where in the Slave's database to locate the data.

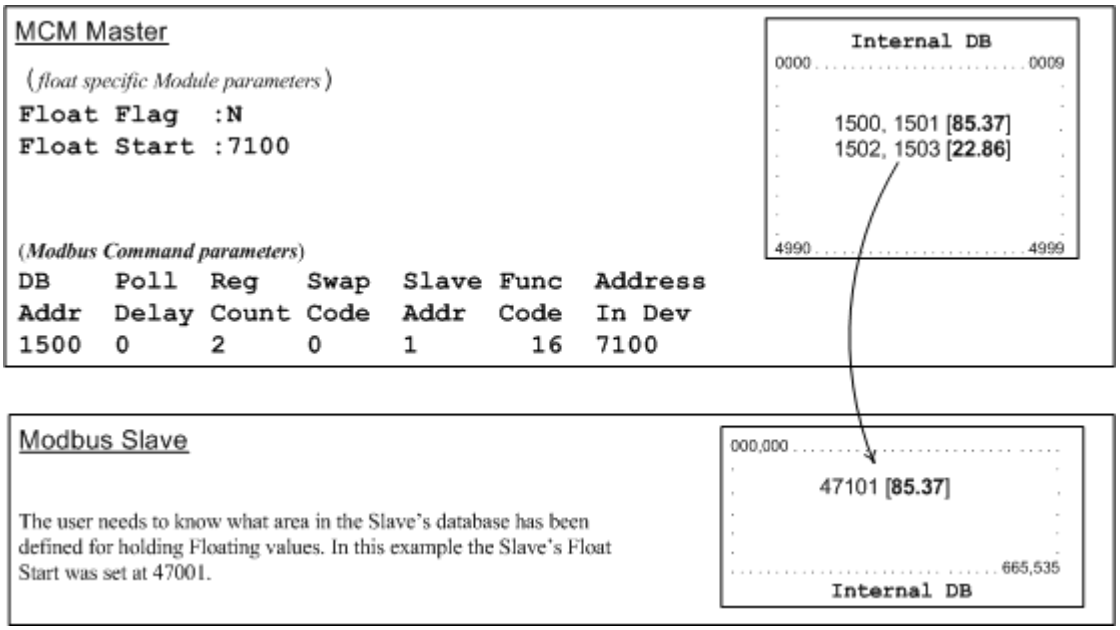
In the above example, the Master's Modbus command to transmit inside the Modbus packet will be as follows.

	Slave address	Function Code	Address in Device	Reg count	Byte Count	Data	
DEC	01	16	7100	2	8	85.37	22.86
HEX	01	10	1B BC	00 02	08	BD 71 42 AA	E1 48 41 B6

In conclusion

The Master's Modbus packet contains the data byte and data word counts that have been doubled from the amount specified by Reg Count due to the Float flag set to Y. Some Slaves look for the byte count in the data packet to know the length of the data to read from the wire. Other slaves know at which byte the data begins and read from the wire the remaining bytes in the packet as the data the Master is sending.

Master is issuing Modbus command with FC 16 (with Float Flag: No) to transfer Float data.


MCM Master
(float specific Module parameters)
Float Flag :N
Float Start :7100
(Modbus Command parameters)
DB Addr | Poll Delay | Reg Count | Swap Code | Slave Addr | Func Code | Address In Dev
1500 | 0 | 2 | 0 | 1 | 16 | 7100
Internal DB
0000
0009
1500, 1501 [85.37]
1502, 1503 [22.86]
4990
4999
Modbus Slave
The user needs to know what area in the Slave's database has been defined for holding Floating values. In this example the Slave's Float Start was set at 47001.
000,000
47101 [85.37]
665,535
Internal DB

Float Flag: "N" tells the Master to ignore the floating values and treat each register data as a data point composed of 1 word, 2 bytes or 16 bits.

Float Start: Ignored.

DB Addr - same as when Float Flag: Y.

Reg Count - Tells the Master how many data points to send to the Slave.

Swap Code - same as when Float Flag: Y.

Func Code - same as when Float Flag: Y.

Addr in Dev - same as when Float Flag: Y as long as the Slave's Float Flag = Y.

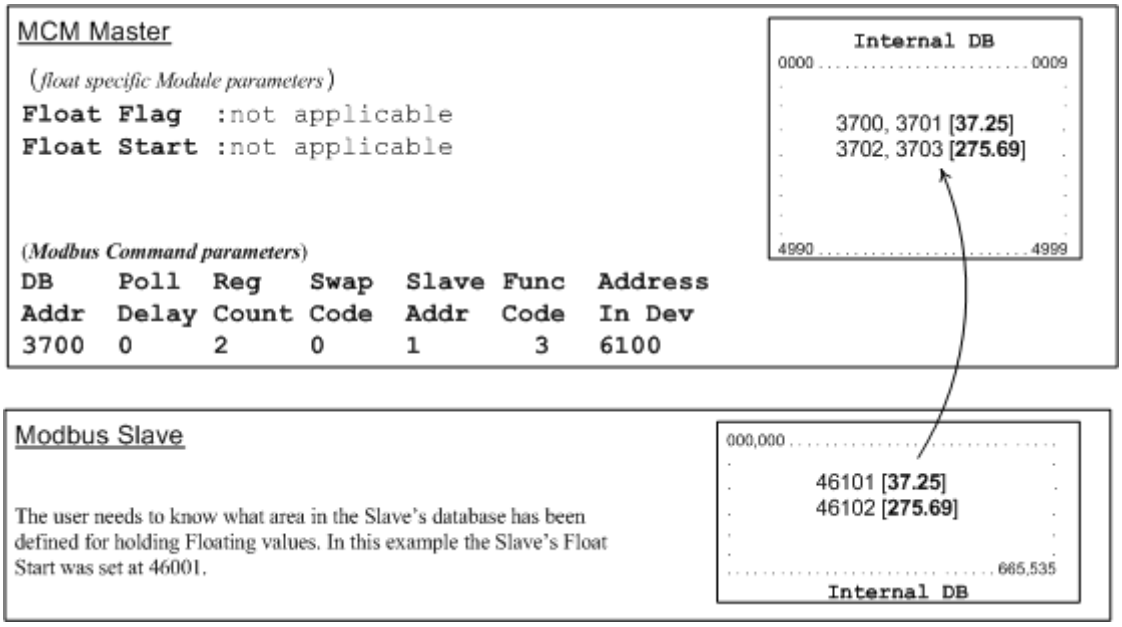
In the above example, the Master's Modbus command to transmit inside the Modbus packet will be as follows.

	Slave address	Function Code	Address in Device	Reg Count	Byte Count	Data
DEC	01	16	7100	2	4	85.37
HEX	01	10	1B BC	00 02	04	BD 71 42 AA

In conclusion

The Master's Modbus packet contains the data byte and data word counts that have NOT been doubled from the amount specified by Reg Count due to the Float Flag set to N. The Slave looks for the byte count in the data packet to know the length of the data to read from the wire. Because of insufficient byte count, some slaves will read only half the data from the Master's transmission. Other slaves will read all 8 bytes in this example because they will know where in the packet the data starts and ignore the byte count parameter inside the Modbus packet.

Master is issuing Modbus command with FC 3 to transfer Float data from Slave.



Float Flag: Not applicable with Modbus Function Code 3.

Float Start: Not applicable with Modbus Function Code 3.

DB Addr - Tells the Master where in its data memory to store the data obtained from the Slave.

Reg Count - Tells the Master how many registers to request from the Slave.

Swap Code - same as above.

Func Code - Tells the Master to read the register values from the Slave. FC3.

Addr in Dev - Tells the Master where in the Slave's database to obtain the data.

In the above example, the Master's Modbus command to transmit inside the Modbus packet will be as follows.

	Slave address	Function Code	Address in Device	Reg count
DEC	01	3	6100	2
HEX	01	03	17 D4	00 02

In the above example the (Enron/Daniel supporting) Slave's Modbus command to transmit inside the Modbus packet will be as follows.

	Slave address	Function Code	Byte Count	Data
DEC	01	3	8	32.75 275.69
HEX	01	03	08	00 00 42 03 D8 52 43 89

In the above example the (a NON-Enron/Daniel supporting) Slave's Modbus command that will be transmitted inside the Modbus packet will be as follows.

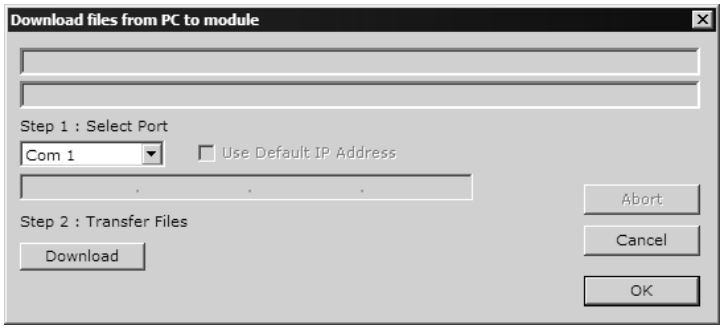
	Slave address	Function Code	Byte Count	Data
DEC	01	3	4	32.75
HEX	01	03	04	00 00 42 03

6.3 Download the Project to the Module

In order for the module to use the settings you configured, you must download (copy) the updated Project file from your PC to the module.

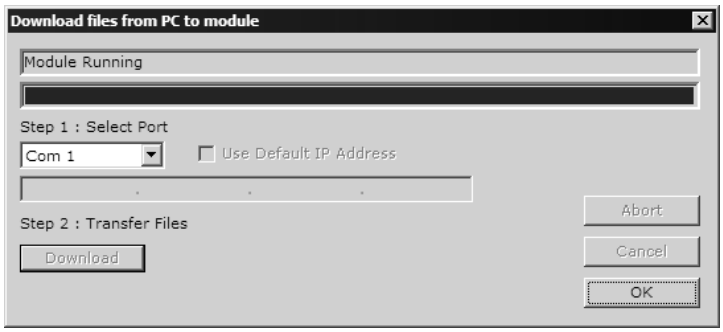
To Download the Project File

1. In the tree view in ProSoft Configuration Builder, click once to select the PTQ-MCM module.
2. Open the **Project menu,** and then choose **Module / Download**. The program will scan your PC for a valid com port (this may take a few seconds). When PCB has found a valid com port, the following dialog box will open.



3. Choose the com port to use from the dropdown list, and then click the Download button.

   The module will perform a platform check to read and load its new settings. When the platform check is complete, the status bar in ProSoft Configuration Builder will be updated with the message *"Module Running*".



6.4 Verification and Troubleshooting

You can now verify that the module is configured properly by viewing parameters that you specified in the configuration file. This is done using the module's Main Menu. If you are not already at the Main menu, press **[Shift][/].**

Use the database menu to verify that data appears in registers that you've mapped. Refer to Diagnostics and Troubleshooting (page 79) for information on accessing information on the operation of the module.

7 Diagnostics and Troubleshooting

In This Chapter

- Reading Status Data from the Module 79
- LED Status Indicators 94
- Error Status Table 95

The module provides information on diagnostics and troubleshooting in the following forms:

- Status data values are transferred from the module to the processor.
- Data contained in the module can be viewed through the Configuration/Debug port attached to a terminal emulator.
- LED status indicators on the front of the module provide information on the module's status.

7.1 Reading Status Data from the Module

The PTQ-MCM module provides the status data in each read block. This data can also be located in the module's database.

7.1.1 Required Hardware

You can connect directly from your computer's serial port to the serial port on the module to view configuration information, perform maintenance, and send (upload) or receive (download) configuration files.

ProSoft Technology recommends the following minimum hardware to connect your computer to the module:

- 80486 based processor (Pentium preferred)
- 1 megabyte of memory
- At least one UART hardware-based serial communications port available. USB-based virtual UART systems (USB to serial port adapters) often do not function reliably, especially during binary file transfers, such as when uploading/downloading configuration files or module firmware upgrades.
- A null modem serial cable.

7.1.2 The Configuration/Debug Menu

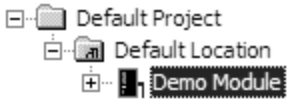
The Configuration and Debug menu for this module is arranged as a tree structure, with the Main Menu at the top of the tree, and one or more sub-menus for each menu command. The first menu you see when you connect to the module is the Main menu.

Because this is a text-based menu system, you enter commands by typing the command letter from your computer keyboard in the diagnostic window in ProSoft Configuration Builder (PCB). The module does not respond to mouse movements or clicks. The command executes as soon as you press the command letter — you do not need to press **[Enter]**. When you type a command letter, a new screen will be displayed in your terminal application.

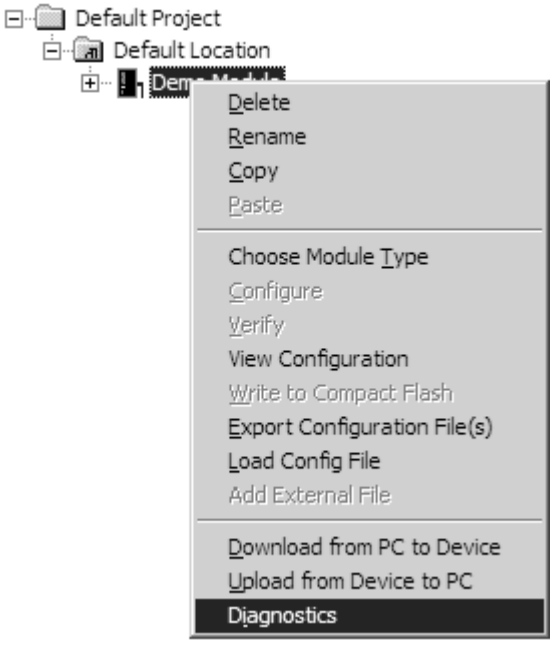
Using the Diagnostic Window in ProSoft Configuration Builder

To connect to the module's Configuration/Debug serial port:

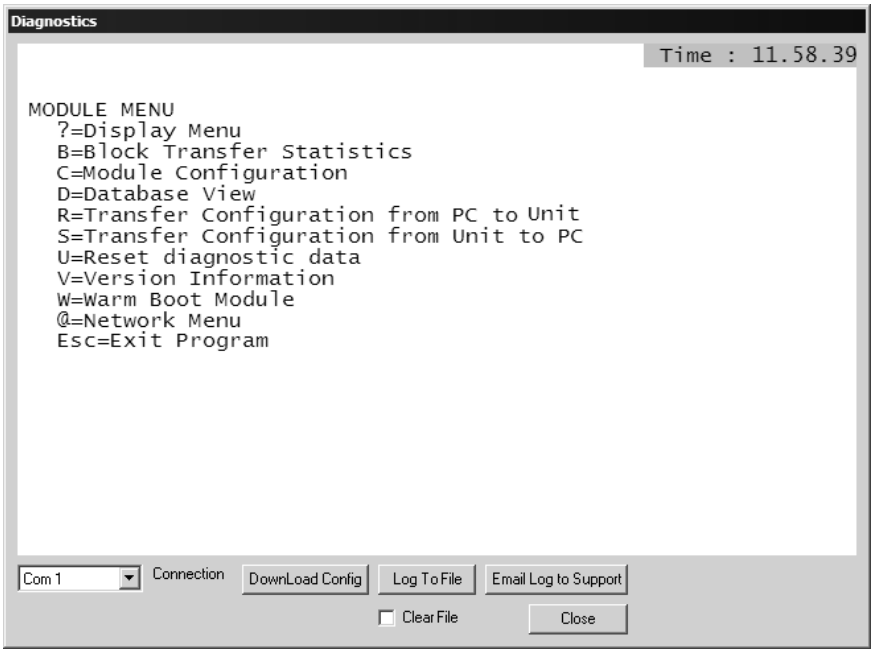
1 Start PCB program with the application file to be tested. Right click over the module icon.



2 On the shortcut menu, choose Diagnostics.



3 This action opens the Diagnostics dialog box. Press "?" to display the Main Menu.



Important: The illustrations of configuration/debug menus in this section are intended as a general guide, and may not exactly match the configuration/debug menus in your own module.

If there is no response from the module, follow these steps:

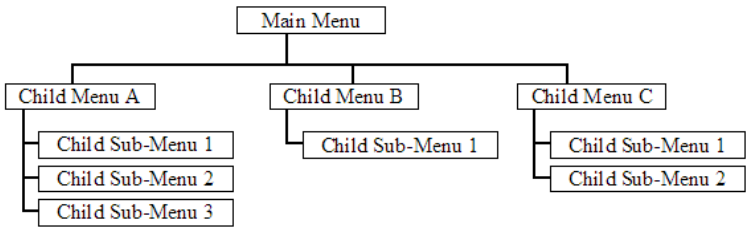
1 Verify that the null modem cable is connected properly between your computer's serial port and the module. A regular serial cable will not work.
2 On computers with more than one serial port, verify that your communication program is connected to the same port that is connected to the module.

If you are still not able to establish a connection, contact ProSoft Technology for assistance.

Navigation

All of the sub-menus for this module contain commands to redisplay the menu or return to the previous menu. You can always return from a sub-menu to the next higher menu by pressing **[M]** on your keyboard.

The organization of the menu structure is represented in simplified form in the following illustration:



The remainder of this section shows you the menus available for this module, and briefly discusses the commands available to you.

Keystrokes

The keyboard commands on these menus are almost always non-case sensitive. You can enter most commands in lower case or capital letters.

The menus use a few special characters (**[?]**, **[-]**, **[+]**, **[@]**) that must be entered exactly as shown. Some of these characters will require you to use the **[Shift]**, **[Ctrl]** or **[Alt]** keys to enter them correctly. For example, on US English keyboards, enter the **[?]** command as **[Shift][/]**.

Also, take care to distinguish capital letter **[I]** from lower case letter **[l]** (L) and number **[1]**; likewise for capital letter **[O]** and number **[0]**. Although these characters look nearly the same on the screen, they perform different actions on the module.

7.1.3 Main Menu

When you first connect to the module from your computer, your terminal screen will be blank. To activate the main menu, press the **[?]** key on your computer's keyboard. If the module is connected properly, the following menu will appear on your terminal screen:

```
PTQ-MCM MENU
 ?=Display Menu
 V=Version Information
 D=Database Menu
 C=Clear diagnostic data
 B=Backplane Menu
 0=Protocol_Serial_MCM 1
 1=Protocol_Serial_MCM 2
 S-Transfer Configuration from Unit to PC
 R=Transfer Configuration from PC to Unit
 W=Warm Boot Module
 Esc=Exit Program
```

Caution: Some of the commands available to you from this menu are designed for advanced debugging and system testing only, and can cause the module to stop communicating with the processor or with other devices, resulting in potential data loss or other failures. Only use these commands if you are specifically directed to do so by ProSoft Technology Technical Support staff. Some of these command keys are not listed on the menu, but are active nevertheless. Please be careful when pressing keys so that you do not accidentally execute an unwanted command.

Viewing Version Information

Press **[V]** to view Version information for the module.

Use this command to view the current version of the software for the module, as well as other important values. You may be asked to provide this information when calling for technical support on the product.

Values at the bottom of the display are important in determining module operation. The Program Scan Counter value is incremented each time a module's program cycle is complete.

Tip: Repeat this command at one-second intervals to determine the frequency of program execution.

Opening the Database Menu

Press **[D]** to open the Database View menu. Use this menu command to view the current contents of the module's database.

Clearing Diagnostic Data

Press **[C]** to clear diagnostic data from the module's memory.

Opening the Backplane Menu

Press **[B]** from the Main Menu to view the Backplane Data Exchange List. Use this command to display the configuration and statistics of the backplane data transfer operations.

Tip: Repeat this command at one-second intervals to determine the number of blocks transferred each second.

Opening the Protocol_Serial_ Menu

Press **[0]** or **[1]** from the Main Menu to open the Protocol_Serial_ menu for Ports 1 and 2.

Use this command to view communication status and statistics for the selected port. This information can be useful for trouble-shooting communication problems.

Transferring the Configuration File from PTQ module to PC

Press **[S]** to receive (download) the configuration file from the module to your PC.

Press **[Y]** to confirm the file transfer, and then follow the instructions on the terminal screen to complete the file transfer process.

After the file has been successfully downloaded, you can open and edit the file to change the module's configuration.

Transferring the Configuration File from PC to PTQ module

Press **[R]** to send (upload) the configuration file from your PC to the module and store the file on the module's Compact Flash Disk.

Press **[Y]** to confirm the file transfer, and then follow the instructions on the terminal screen to complete the file transfer process.

After the file has been successfully downloaded, the module will restart the program and load the new configuration information. Review the new configuration using menu commands **[6]** and **[0]** to verify that the module is configured correctly.

Warm Booting the Module

Caution: Some of the commands available to you from this menu are designed for advanced debugging and system testing only, and can cause the module to stop communicating with the processor or with other devices, resulting in potential data loss or other failures. Only use these commands if you are specifically directed to do so by ProSoft Technology Technical Support staff. Some of these command keys are not listed on the menu, but are active nevertheless. Please be careful when pressing keys so that you do not accidentally execute an unwanted command.

Press **[W]** from the Main Menu to warm boot (restart) the module. This command will cause the program to exit and reload, refreshing configuration parameters that must be set on program initialization. Only use this command if you must force the module to re-boot.

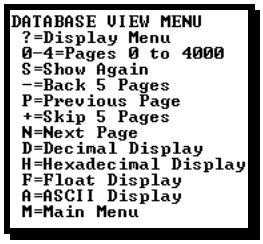
Exiting the Program

Caution: Some of the commands available to you from this menu are designed for advanced debugging and system testing only, and can cause the module to stop communicating with the processor or with other devices, resulting in potential data loss or other failures. Only use these commands if you are specifically directed to do so by ProSoft Technology Technical Support staff. Some of these command keys are not listed on the menu, but are active nevertheless. Please be careful when pressing keys so that you do not accidentally execute an unwanted command.

Press **[Esc]** to restart the module and force all drivers to be loaded. The module will use the configuration stored in the module's Flash ROM to configure the module.

7.1.4 Modbus Database View

Press **[D]** to open the Modbus Database View menu. Use this command to view the module's internal database values. Press **[?]** to view a list of commands on this menu.



All data contained in the module's database is available for viewing using the commands. Refer to Modbus Protocol Specification (page 114) for information on the structure of Modbus messages. Each option available on the menu is discussed in the following topics.

Viewing Register Pages

To view sets of register pages, use the keys described below:

Command	Description
[0]	Display registers 0 to 99
[1]	Display registers 1000 to 1099
[2]	Display registers 2000 to 2099

And so on. The total number of register pages available to view depends on your module's configuration.

Redisplaying the Current Page

Press **[S]** to display the current page of data.

Moving Back Through 5 Pages of Registers

Press **[-]** from the Database View menu to skip back to the previous 500 registers of data.

Viewing the Previous 100 Registers of Data

Press **[P]** from the Database View menu to display the previous 100 registers of data.

Skipping 500 Registers of Data

Hold down **[Shift]** and press **[=]** to skip forward to the next 500 registers of data.

Viewing the Next 100 Registers of Data

Press **[N]** from the Database View menu to select and display the next 100 registers of data.

Viewing Data in Decimal Format

Press **[D]** to display the data on the current page in decimal format.

Viewing Data in Hexadecimal Format

Press **[H]** to display the data on the current page in hexadecimal format.

Viewing Data in Floating Point Format

Press **[F]** from the Database View menu. Use this command to display the data on the current page in floating point format. The program assumes that the values are aligned on even register boundaries. If floating-point values are not aligned as such, they are not displayed properly.

Viewing Data in ASCII (Text) Format

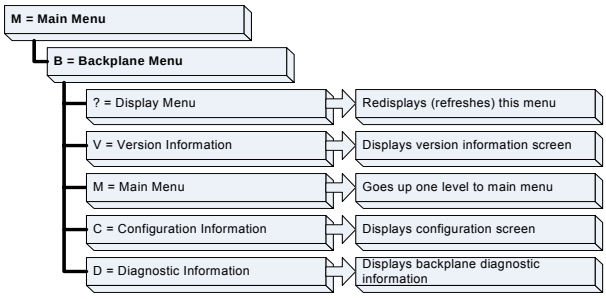
Press **[A]** to display the data on the current page in ASCII format. This is useful for regions of the database that contain ASCII data.

Returning to the Main Menu

Press **[M]** to return to the Main Menu.

7.1.5 Backplane Menu

Press **[B]** from the Main Menu to view the Backplane Data Exchange List. Use this command to display the configuration and statistics of the backplane data transfer operations. Press **[?]** to view a list of commands available on this menu.


M = Main Menu
B = Backplane Menu
? = Display Menu
Redisplays (refreshes) this menu
V = Version Information
Displays version information screen
M = Main Menu
Goes up one level to main menu
C = Configuration Information
Displays configuration screen
D = Diagnostic Information
Displays backplane diagnostic information

Redisplaying the Menu

Press **[?]** to display the current menu. Use this command when you are looking at a screen of data, and want to view the menu choices available to you.

Viewing Version Information

Press **[V]** to view Version information for the module.

Use this command to view the current version of the software for the module, as well as other important values. You may be asked to provide this information when calling for technical support on the product.

Values at the bottom of the display are important in determining module operation. The Program Scan Counter value is incremented each time a module's program cycle is complete.

Tip: Repeat this command at one-second intervals to determine the frequency of program execution.

Returning to the Main Menu

Press **[M]** to return to the Main Menu.

Viewing Configuration Information

Press **[C]** to view configuration information for the selected port, protocol, driver or device.

Viewing Backplane Diagnostic Information

Press **[D]** to view Backplane Diagnostic information.

Use this command to display the configuration and statistics of the backplane data transfer operations between the module and the processor. The information on this screen can help determine if there are communication problems between the processor and the module.

Tip: Repeat this command at one-second intervals to determine the number of blocks transferred each second

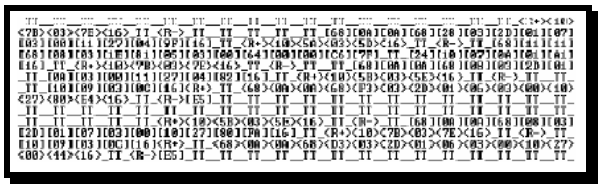
7.1.6 Data Analyzer

The data analyzer mode allows you to view all bytes of data transferred on each port. Both the transmitted and received data bytes are displayed. Use of this feature is limited without a thorough understanding of the protocol.

Note: The Port selection commands on the Data Analyzer menu differs very slightly in different modules, but the functionality is basically the same. Use the illustration above as a general guide only. Refer to the actual data analyzer menu on your module for the specific port commands to use.
Important: When in analyzer mode, program execution will slow down. Only use this tool during a troubleshooting session. Before disconnecting from the Config/Debug port, please press [S] to stop the data analyzer, and then press [M] to return to the main menu. This action will allow the module to resume its normal high speed operating mode.

Analyzing Data for the first application port

Press **[1]** to display I/O data for the first application port in the Data Analyzer. The following illustration shows an example of the Data Analyzer output.



Analyzing Data for the second application port

Press **[2]** to display I/O data for the second application port in the Data Analyzer.

Displaying Timing Marks in the Data Analyzer

You can display timing marks for a variety of intervals in the data analyzer screen. These timing marks can help you determine communication-timing characteristics.

Key	Interval
[5]	1 milliseconds ticks
[6]	5 milliseconds ticks
[7]	10 milliseconds ticks
[8]	50 milliseconds ticks
[9]	100 milliseconds ticks
[0]	Turn off timing marks

Removing Timing Marks in the Data Analyzer

Press **[0]** to turn off timing marks in the Data Analyzer screen.

Viewing Data in Hexadecimal Format

Press **[H]** to display the data on the current page in hexadecimal format.

Viewing Data in ASCII (Text) Format

Press **[A]** to display the data on the current page in ASCII format. This is useful for regions of the database that contain ASCII data.

Starting the Data Analyzer

Press **[B]** to start the data analyzer. After the key is pressed, all data transmitted and received on the currently selected port will be displayed. An example display is shown below:

The Data Analyzer displays the following special characters:

Character	Definition
[ ]	Data enclosed in these characters represent data received on the port.
< >	Data enclosed in these characters represent data transmitted on the port.
<R+>	These characters are inserted when the RTS line is driven high on the port.
<R->	These characters are inserted when the RTS line is dropped low on the port.
<CS>	These characters are displayed when the CTS line is recognized high.
_TT_	These characters are displayed when the timing mark interval has been reached. This parameter is user defined.

Stopping the Data Analyzer

Press **[S]** to stop the data analyzer. Use this option to freeze the display so the data can be analyzed. To restart the analyzer, press **[B]**.

Important: When in analyzer mode, program execution will slow down. Only use this tool during a troubleshooting session. Before disconnecting from the Config/Debug port, please press [S] to stop the data analyzer, and then press [M] to return to the main menu. This action will allow the module to resume its normal high speed operating mode.

Returning to the Main Menu

Press **[M]** to return to the Main Menu.

7.1.7 Data Analyzer Tips

From the main menu, press **[A]** for the "Data Analyzer". You should see the following text appear on the screen:

```
Data Analyzer Mode Selected
```

After the "Data Analyzer" mode has been selected, press **[?]** to view the Data Analyzer menu. You will see the following menu:

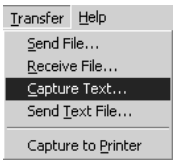
```
DATA ANALYZER VIEW MENU
 ?=Display Menu
 1=Select Port 1
 2=Select Port 2
 5=1 mSec Ticks
 6=5 mSec Ticks
 7=10 mSec Ticks
 8=50 mSec Ticks
 9=100 mSec Ticks
 0=No mSec Ticks
 H=Hex Format
 A=ASCII Format
 B=Start
 S=Stop
 M=Main Menu

 Port = 1, Format=HEX, Tick=10
```

From this menu, you can select the "Port", the "format", and the "ticks" that you can display the data in.

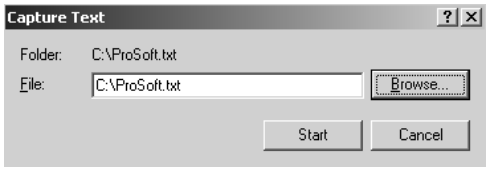
For most applications, HEX is the best format to view the data, and this does include ASCII based messages (because some characters will not display on HyperTerminal and by capturing the data in HEX, we can figure out what the corresponding ASCII characters are supposed to be).

The Tick value is a timing mark. The module will print a _TT for every xx milliseconds of no data on the line. Usually 10milliseconds is the best value to start with.

After you have selected the Port, Format, and Tick, we are now ready to start a capture of this data. The easiest way to do so is to go up to the top of you HyperTerminal window, and do a **Transfer / Capture Text** as shown below:



After selecting the above option, the following window will appear:



Next name the file, and select a directory to store the file in. In this example, we are creating a file ProSoft.txt and storing this file on our root C: drive. After you have done this, press the Start button.

Now you have everything that shows up on the HyperTerminal screen being logged to a file called ProSoft.txt. This is the file that you will then be able to email to ProSoft Technical Support to assist with issues on the communications network.

To begin the display of the communications data, you will then want to press 'B' to tell the module to start printing the communications traffic out on the debug port of the module. After you have pressed 'B', you should see something like the following:

```
[03][00][04][00][05][00][06][00][07][00][08][00][09][FB][B7]_TT__TT_<R+><01><02>
<00><00><00><0A><F8><0D><R->_TT__TT__TT_[01][02][02][00][00][B9][B8]_TT__TT_<R+>
<01><03><00><00><00><0A><C5><CD><R->_TT__TT_[01][03][14][00][00][00][01][00]_TT_
[02][00][03][00][04][00][05][00][06][00][07][00][08][00][09][CD][51]_TT__TT_<R+>
<01><01><00><00><00><A0><3C><72><R->_TT__TT_[01][01][14][00][00][01][00][02]_TT_
[00][03][00][04][00][05][00][06][00][07][00][08][00][09][00][B7][52]_TT__TT_<R+>
<01><04><00><00><00><0A><70><0D><R->_TT__TT_[01][04][14][00][00][00][01][00]_TT_
[02][00][03][00][04][00][05][00][06][00][07][00][08][00][09][FB][B7]_TT__TT_<R+>
<01><02><00><00><00><0A><F8><0D><R->_TT__TT__TT_[01][02][02][00][00][B9][B8]_TT_
_TT_<R+><01><03><00><00><00><0A><C5><CD><R->_TT__TT_[01][03][14][00][00][00][01]
[00]_TT_[02][00][03][00][04][00][05][00][06][00][07][00][08][00][09][CD][51]_TT_
_TT_<R+><01><01><00><00><00><A0><3C><72><R->_TT__TT__TT_[01][01][14][00][00][01]
[00][02]_TT_[00][03][00][04][00][05][00][06][00][07][00][08][00][09][00][B7][52]
_TT__TT_<R+><01><04><00><00><00><0A><70><0D><R->_TT__TT_[01][04][14][00][00][00]
[01][00]_TT_[02][00][03][00][04][00][05][00][06][00][07][00][08][00][09][FB][B7]
_TT__TT_<R+><01><02><00><00><00><0A><F8><0D><R->_TT__TT_[01][02][02][00][00][B9]
[B8]_TT__TT_<R+><01><03><00><00><00><0A><C5><CD><R->_TT__TT_[01][03][14][00][00]
[00][01][00]_TT_[02][00][03][00][04][00][05][00][06][00][07][00][08][00][09][CD]
[51]_TT__TT_<R+><01><01><00><00><00><A0><3C><72><R->_TT__TT__TT_[01][01][14][00]
[00][01][00][02]_TT_[00][03][00][04][00][05][00][06][00][07][00][08][00][09][00]
[B7][52]_TT__TT_<R+><01><04><00><00><00><0A><70><0D><R->_TT__TT_[01][04][14][00]
[00][00][01][00]_TT_[02][00][03][00][04][00][05][00][06][00][07][00][08][00][09]
[FB][B7]_TT__TT_<R+><01><02><00><00><00><0A><F8><0D><R->_TT__TT__TT_[01][02][02]
[00][00][B9][B8]_TT__TT_<R+><01><03><00><00><00><0A><C5><CD><R->_TT__TT_
```

The <R+> means that the module is transitioning the communications line to a transmit state.

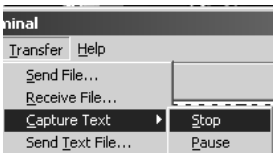
All characters shown in <> brackets are characters being sent out by the module.

The <R-> shows when the module is done transmitting data, and is now ready to receive information back.

And finally, all characters shown in the [ ] brackets is information being received from another device by the module.

After taking a minute or two of traffic capture, you will now want to stop the "Data Analyzer". To do so, press the 'S' key, and you will then see the scrolling of the data stop.

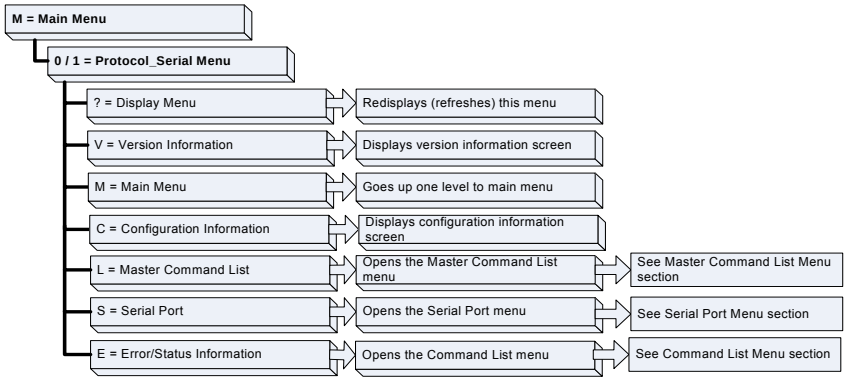
When you have captured the data you want to save, open the Transfer menu and choose Capture Text. On the secondary menu, choose Stop.



You have now captured, and saved the file to your PC. This file can now be used in analyzing the communications traffic on the line, and assist in determining communication errors.

7.1.8 Protocol Serial Menu

Press **[0]** or **[1]** to view protocol serial information for ports 1 and 2, respectively. Use this command to view a variety of error and status screens for the port. Press **[?]** to view a list of commands available on this menu.


M = Main Menu
0 / 1 = Protocol_Serial Menu
? = Display Menu
Redisplays (refreshes) this menu
V = Version Information
Displays version information screen
M = Main Menu
Goes up one level to main menu
C = Configuration Information
Displays configuration information screen
L = Master Command List
Opens the Master Command List menu
See Master Command List Menu section
S = Serial Port
Opens the Serial Port menu
See Serial Port Menu section
E = Error/Status Information
Opens the Command List menu
See Command List Menu section

Redisplaying the Menu

Press **[?]** to display the current menu. Use this command when you are looking at a screen of data, and want to view the menu choices available to you.

Viewing Version Information

Press **[V]** to view Version information for the module.

Use this command to view the current version of the software for the module, as well as other important values. You may be asked to provide this information when calling for technical support on the product.

Values at the bottom of the display are important in determining module operation. The Program Scan Counter value is incremented each time a module's program cycle is complete.

Tip: Repeat this command at one-second intervals to determine the frequency of program execution.

Returning to the Main Menu

Press **[M]** to return to the Main Menu.

Viewing Configuration Information

Press **[C]** to view configuration information for the selected port, protocol, driver or device.

Opening the Command List Menu

Press **[L]** to open the Command List menu. Use this command to view the configured command list for the module.

Opening the Serial Port Menu

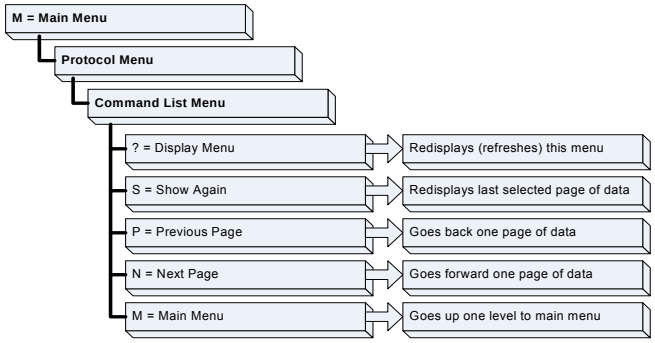
Press **[S]** to open the Serial Port menu. Use this command to view and change additional serial port driver settings.

Viewing Error and Status Data

Press **[E]** to display the error/status data for the module.

7.1.9 Master Command Error List Menu

Use this menu to view the command error list for the module. Press **[?]** to view a list of commands available on this menu.


M = Main Menu
Protocol Menu
Command List Menu
? = Display Menu
Redisplays (refreshes) this menu
S = Show Again
Redisplays last selected page of data
P = Previous Page
Goes back one page of data
N = Next Page
Goes forward one page of data
M = Main Menu
Goes up one level to main menu

Redisplaying the Current Page

Press **[S]** to display the current page of data.

Viewing the Previous 20 Commands

Press **[-]** to display data for the previous 20 commands.

Viewing the Previous Page of Commands

Press **[P]** to display the previous page of commands.

Viewing the Next 20 Commands

Press **[+]** to display data for the next 20 commands.

Viewing the Next Page of Commands

Press **[N]** to display the next page of commands.

Returning to the Main Menu

Press **[M]** to return to the Main Menu.

7.2 LED Status Indicators

The LEDs will indicate the module's operating status as follows:

ProTalk Module	Color	Status	Indication
DEBUG	Green	On	Data is being transferred between the module and a remote terminal using the Configuration/Debug port.
		Off	No data is being transferred on the Configuration/Debug port.
PRT1	Green	On	Port is communicating.
		Off	Port is not communicating.
PRT2	Green	On	Port is communicating.
		Off	Port is not communicating.
ERR1 CFG Fail	Red	Off	The PTQ-MCM is working normally.
		On	The PTQ-MCM module program has recognized an application error. This LED will also be turned on if any command presents an error.
ERR2 P1 Fail	Red	On	Time-out. The post sent a Modbus command but did not receive a response from the slave.
		Off	Normal Operation
ERR3 P2 Fail	Red	On	Time-out. The post sent a Modbus command but did not receive a response from the slave.
		Off	Normal Operation
Active	Green	On	The LED is on when the module recognizes a processor and is able to communicate if the [Backplane Data Exchange] section specifies data transfer commands.
		Off	The LED is off when the module is unable to speak with the processor. The processor either absent or not running.
BAT Low	Red	Off	The battery voltage is OK and functioning.
		On	The battery voltage is low or the battery is not present. The battery LED will illuminate briefly upon the first installation of the module or if the unit has been un-powered for an extended period of time. This behavior is normal, however should the LED come on in a working installation please contact ProSoft Technology.
E-Link	Green	N/A	Not Used
		N/A	Not Used
E-Data	Green	N/A	Not Used
		N/A	Not Used

If your module is not operating, and the status LEDs are not illustrated in the table above, please call ProSoft Technology for technical assistance.

7.3 Error Status Table

The program maintains an error/status table that is transferred to the processor in each read block. You can use the error/status data to determine the "health" of the module. The data block structure is describe in the Status Data Definition section. The errors generated by the module are displayed in the following tables.

7.3.1 Standard Modbus Protocol Errors

Code	Description
1	Illegal Function
2	Illegal Data Address
3	Illegal Data Value
4	Failure in Associated Device
5	Acknowledge
6	Busy, Rejected Message

7.3.2 Module Communication Error Codes

Code	Description
-1	CTS modem control line not set before transmit
-2	Timeout while transmitting message
-11	Timeout waiting for response after request
253	Incorrect slave address in response
254	Incorrect function code in response
255	Invalid CRC/LRC value in response

7.3.3 Command List Entry Errors

Code	Description
-41	Invalid enable code
-42	Internal address > maximum address
-43	Invalid node address (< 0 or > 255)
-44	Count parameter set to 0
-45	Invalid function code
-46	Invalid swap code

8 Reference

In This Chapter

- Product Specifications 97
- Functional Overview 99
- Status Data Definition 110
- Configuration Data 112
- Port Specific Information 114
- Modbus Protocol Specification 114
- Frequently Asked Questions 125

8.1 Product Specifications

The ProTalk Modbus Master/Slave Communication Module allows Schneider Electric Quantum processors to interface easily with other Modbus protocol compatible devices and hosts. The data transfer from the Quantum processor is asynchronous from the actions on the Modbus network.

Compatible devices include not only Modicon PLCs (which all support the Modbus protocol) but also a wide assortment of end devices.

8.1.1 Features and Benefits

Many host SCADA applications support the Modbus protocol, while devices commonly supporting the protocol include several PLCs, as well as many other third party devices in the marketplace. (For a partial list of devices that speak Modbus, please visit the ProSoft Tested section of the ProSoft Technology web site).

8.1.2 General Specifications

- Single Slot - Quantum backplane compatible
- The module is recognized as an Options module and has access to PLC memory for data transfer
- Configuration data is stored in non-volatile memory in the ProTalk module
- Configuration software for Microsoft Windows XP, 2000 and NT is included with the module.
- Up to six modules can be placed in a rack
- Local rack - The module must be placed in the same rack as processor.
- Compatible with common Quantum / Unity programming tools.
  - UnityPro XL
  - Concept
  - ProWORX
- Quantum data types supported: 3x, 4x
- High speed data transfer across backplane provides quick data update times.
- Sample ladder file available.

8.1.3 Hardware Specifications

Specification	Value
Backplane Current Load	800 mA @ 5 V
Operating Temperature	0 to 60°C (32 to 140°F)
Storage Temperature	-40 to 85°C (-40 to 185°F)
Relative Humidity	5% to 95% (non-condensing)
Vibration	Sine vibration 4-100 Hz in each of the 3 orthogonal axes
Shock	30G, 11 mSec. in each of the 3 orthogonal axes
LED Indicators	Module Status Backplane Transfer Status Serial Port Activity LED Serial Activity and Error LED Status
Configuration Serial Port (PRT1)	DB-9M PC Compatible RS-232 only No hardware handshaking
Application Serial Ports	(PRT2, PRT3) DB-9M PC Compatible RS-232/422/485 jumper selectable RS-422/485 screw termination included RS-232 handshaking configurable 500V Optical isolation from backplane

8.1.4 Functional Specifications

- Support for the storage and transfer of up to 5000 registers to/from the Quantum processor's data files
- User-definable module memory usage
- Two ports to emulate any combination of Modbus master or slave device
- Supports Enron version of Modbus protocol for floating point data transactions

Configurable parameters include:

Parameter	Description
Protocol	RTU or ASCII
Baud Rate	110 to 115,200
Parity	None, Odd and Even
Data Bits	5 to 8
Stop Bits	1 or 2
RTS On and Off Timing	0 to 65535 milliseconds
Minimum Response Delay	0 to 65535 milliseconds
Use of CTS Modem Line	Yes or No
Floating-Point Support	

Slave Specifications

The PTQ-MCM module accepts Modbus function code commands of 1, 2, 3, 4, 5, 6, 8, 15, 16, 17, 22 and 23 from an attached Modbus Master unit. A port configured as a Modbus slave permits a remote master to interact with all data contained in the module. This data can be derived from other Modbus slave devices on the network, through a master port, or from the Quantum processor.

Master Specifications

A port configured as a virtual Modbus Master device on the PTQ-MCM module actively issues Modbus commands to other nodes on the Modbus network. One hundred commands are supported on each port. Additionally, the master ports have an optimized polling characteristic that polls slaves with communication problems less frequently. The Quantum processor can be programmed to control the activity on the port by actively selecting commands from the command list to execute or issuing commands directly from the ladder logic.

8.2 Functional Overview

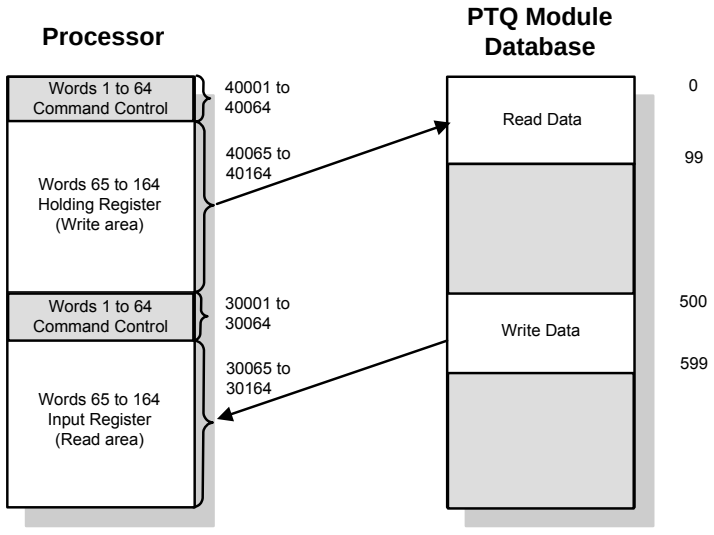
8.2.1 Backplane Data Transfer

The current version of the PTQ-MCM backplane driver (version 2.10 or newer), uses a Large I/O model, which differs from previous versions of the backplane driver in that it transfers all of the data in the Read and Write databases between the module and the processor on every scan.

The [Backplane Configuration] section of the configuration file defines the starting registers for read and write operations, as well as the number of registers to use for each data area.

```
[Backplane Configuration]

Failure Flag Count           : 0    #
Error/Status Block Pointer   : 3000 #Number of register to store error and
                                     #status
```

The values in the example configuration file section above are illustrated in the following diagram.



The module transfers the entire read and write areas at the end of every processor scan. The module will hold the processor scan for a certain period of time, which allows the module to transfer the entire read and write areas. This means that the larger the read and write areas, the longer the processor scan time will be. Refer to Module Performance for more detailed information on determining scan times.

Note: The diagram above shows the memory addresses for a Quantum / Unity processor. If you are deploying the PTQ-MCM with a Unity processor, substitute %MW for read only data, and %IW for read/write data.

8.2.2 Special Functions

Command control blocks are special blocks used to control the module or request special data from the module. The current version of the software supports four command control blocks: event command control, command control, warm boot and cold boot.

8.2.3 Event Command

Event command control blocks send Modbus commands directly from the ladder logic to one of the master ports. The format for these blocks is displayed below:

Write Block - Event Command

Offset	Description	Length
0	1000 to 1247 or 2000 to 2247	1
1	Internal DB Address	1
2	Point Count	1
3	Swap Code	1
4	Function Code	1
5	Destination Code	1
6 to 63	Spare	1

The block number defines the Modbus port to be considered. Block 1000 commands are directed to Port 1, and block 2000 commands are directed to Port 2. The slave address is represented in the block number in the range of 0 to 255. The sum of these two values determines the block number. The parameters passed with the block are used to construct the command.

- The **Internal DB Address** parameter specifies the module's database location to associate with the command.
- The **Point Count** parameter defines the number of registers for the command.
- The **Swap Code** is used to change the word or byte order.
- The **Node Address** parameter is used to define the device on the Modbus network to consider.
- The **Function Code** parameter is one of those defined in the section [Modbus Port 1 Commands].

The parameter fields in the block should be completed as required by the selected function code. Each command has its own set of parameters. When the block is received, the module will process it and place the command in the command queue. The module will respond to each event command block with a read block with the following format:

Read Block: Event Command

Offset	Description	Length
0	0	1
1	Write Block ID	1
2	0=Fail, 1=Success	1
3 to n	Spare	

Word two of the block can be used by the processor logic to determine if the command was added to the command queue of the module. The command will only fail if the command queue for the port is full (100 commands for each queue) or the command requested is invalid.

8.2.4 Command Control Block

The first 64 words of each block are reserved for command control. Each command control block has a Block ID number (shown in parentheses below) that identifies the command control instruction. The PTQ-MCM module supports the following command control blocks:

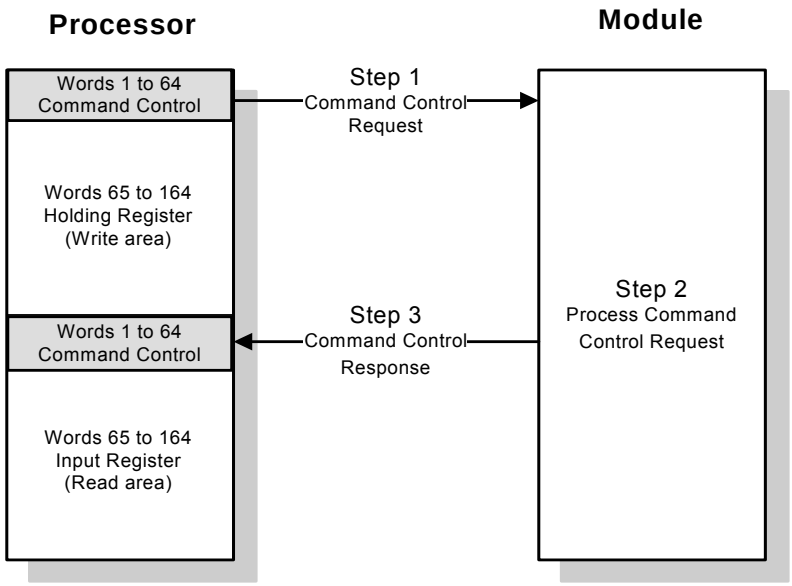
- Event Command (1000 to 1247 Port 1; 2000 to 2247 Port 2)
- Command Control (5001 to 5006 Port 1; 5101 to 5106 Port 2)
- Warm Boot (9998)
- Cold Boot (9999)

The value in word 0 of this 64 word block is the block sequence number. This number identifies whether the contents of the block have changed. This is the actual trigger to send the control request to the module.

Processor logic must be built to handle the command control functionality. The logic would typically follow these steps:

1. Move the block request to output command control area.
2. Move a new value to the output block sequence number.
3. If the input block sequence number equals the output block sequence number + 1, copy the block response to appropriate variables in the module's memory.

Note: Command Control blocks are not copied to the module database. You must define variables in the module's main memory, and use processor logic to process the command control request.



The following table shows the contents of the command control area when a command control block such as 9998 (Warm Boot module) is issued.

Note: The diagram above shows the memory addresses for a Quantum / Unity processor. If you are deploying the PTQ-MCM with a Unity processor, substitute %MW for read only data, and %IW for read/write data.
Note: The processor memory locations in the example tables below use the 3x register start and 4x register start values defined in Backplane Data Transfer (page 99). You can configure any valid 3x and 4x start address that is not used by other processes.

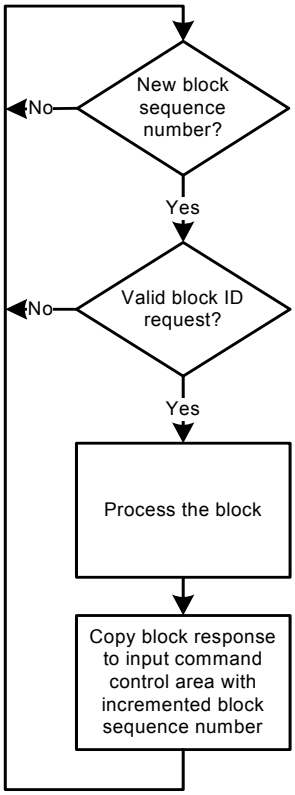
Command Control Word	Description
40001	Output sequence number
40002	Block ID
40003	Block request word 1
40004	Block request word 2
40005	Block request word 3
…	…
40064	Block request word 62

The following table shows the results of the PTQ-MCM response to the command control block.

Command Control Word	Description
30001	Input sequence number
30002	Block ID
30003	Block response word 1

Command Control Word	Description
30004	Block response word 2
30005	Block response word 3
…	…
30064	Block response word 62

The module recognizes that there is a new block request when it identifies that the block sequence number has changed. if the block ID is valid, the module will process the block and copy the response to the input command control area (3x for Quantum / Unity or %IW for Unity). The module will increment the block sequence number by one, as shown in the following illustration.



Write Block - Command Control

Offset	Description	Length
0	5001 to 5006 or 5101 to 5106	1
1	Command index	1
2	Command index	1
3	Command index	1
4	Command index	1
5	Command index	1
6	Command index	1
7 to 63	Spare	

Blocks in the range of 5001 to 5006 are used for Port 1, and blocks in the range of 5101 to 5106 are used for Port 2. The last digit in the block code defines the number of commands to process in the block. For example, a block code of 5003 contains 3 command indexes that are to be used with Port 1. The Command index parameters in the block have a range of 0 to 99 and correspond to the master command list entries.

The module responds to a command control block with a block containing the number of commands added to the command queue for the port. The format of the block is displayed below:

Read Block: Command Control

Offset	Description	Length
0	5000 to 5006 or 5100 to 5106	1
1	Write Block ID	1
2	Number of commands added to command queue	1
3 to (n+1)	Spare	

8.2.5 Warm Boot

This block is sent from the processor to the module when the module is required to perform a warm-boot (software reset) operation. The structure of the control block is shown below:

Offset	Description	Length
0	9998	1
1 to 63	Spare	63

8.2.6 Cold Boot

This block is sent from the processor to the module when the module is required to perform the cold boot (hardware reset) operation. This block is sent to the module when a hardware problem is detected by the processor logic that requires a hardware reset. The structure of the control block is shown below:

Offset	Description	Length
0	9999	1
1 to 63	Spare	63

8.2.7 Pass-Thru Control Blocks

Overview

Caution: This feature allows a remote modbus master to directly update the processor memory. The user should carefully read this section before considering this feature in order to avoid transferring data to unexpected memory areas in the Quantum processor.

The pass-thru operation consists of the slave port receiving a write command from the master and passing the data directly to the processor without updating the PTQ-MCM internal database.

The module supports two pass-thru modes: block pass-thru and direct pass-thru. The module also supports swap byte option for each mode.

During block pass-thru operation the module generates a specific block to the processor when it receives a write command from the master. The module uses different block IDs depending on the type of write command received (modbus functions 5,6,15 and 16). The user should implement processor logic to handle the pass-thru block once it is received. The block contains relevant information such as data count, destination address, and data

During direct pass-thru operation the module passes the data received from a write command to the processor. The module will use the destination address from the modbus command to select the start memory address where the data will be copied to.

Pass-Thru Configuration

Each port can be configured for pass-thru functionality.

```
Type: 0 #Type of port (0=Master, 1=Slave, 2=Block Pass-Thru,
     #3=Block Pass-Thru, 4=Direct Pass-Thru, 5=Direct Pass-Thru Swap)
```

The following codes are supported for Pass-Thru:

Code	Description	Modbus Functions Supported
2	Block Pass-Thru	5, 6, 15, 16
3	Block Pass-Thru with Byte Swap	6, 16
4	Direct Pass-Thru	6, 16
5	Direct Pass-Thru with Byte Swap	6, 16

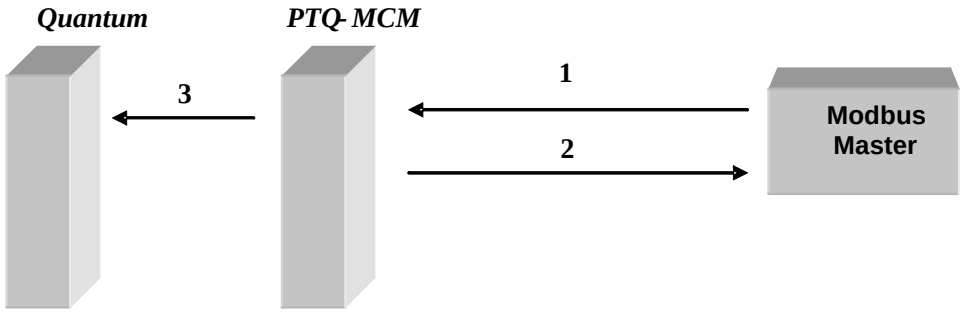
The maximum number of bits supported for modbus function 15 is 800 bits. The maximum number of words supported for modbus function 16 is 125 words.

Direct Pass-Thru

When a PTQ-MCM port is configured for pass-thru operation it will work as a modbus slave transferring the data received directly to the processor memory (holding register memory). The configuration of backplane commands is not required to transfer data from the module to the processor during direct pass-thru operation. This feature does not require any processor program to receive the data from the module.

The module uses the destination address received from the modbus command to determine the memory location in the processor where the data will be written to.

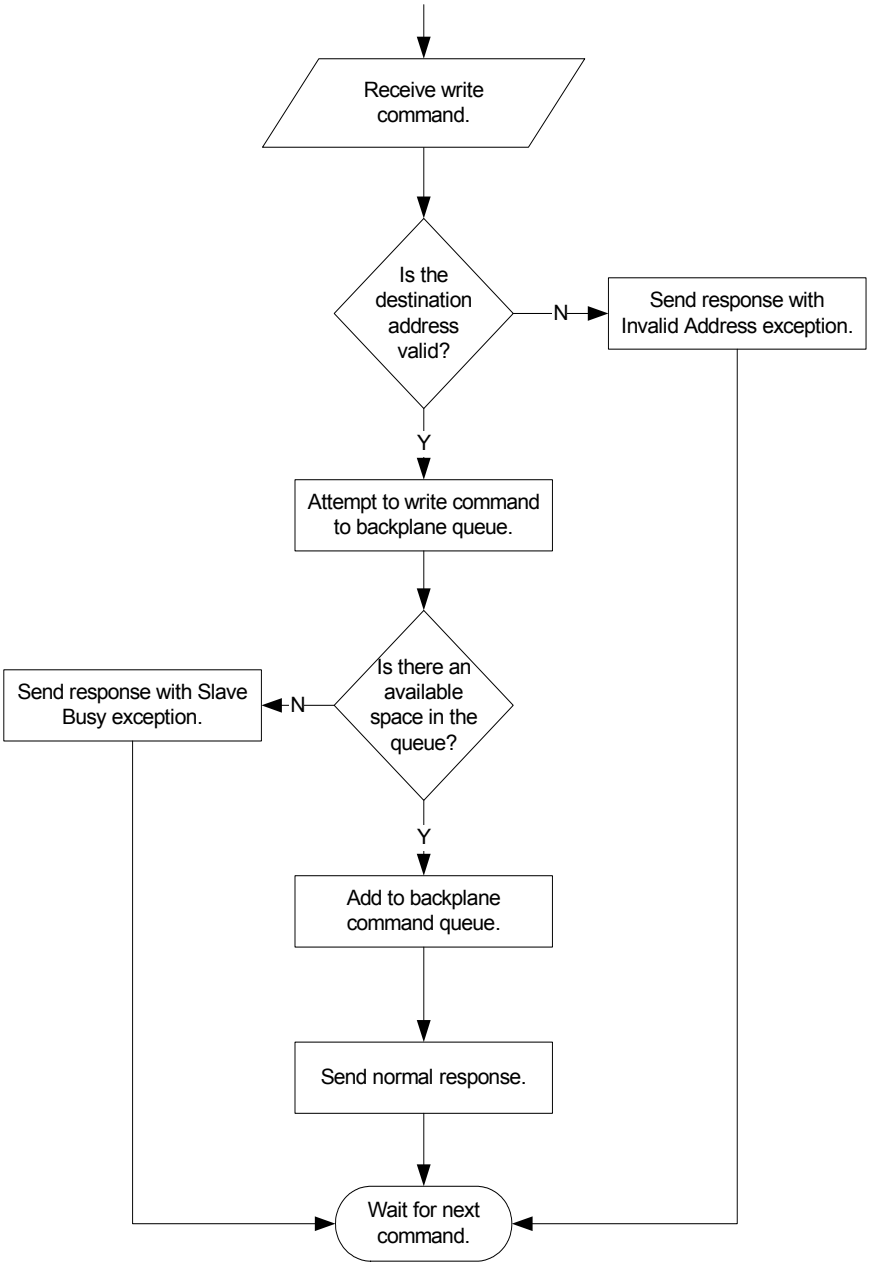
The following illustration shows the three basic steps performed during the pass-thru operation.



The module receives a modbus write command. For example:

1 Modbus Function: Preset Multiple Registers (function 16)
2 Count: 10 words
3 Destination Address: 500
4 The module verifies that the destination address is a valid address in the processor. If not, it sends a response with an exception code (illegal data address). If the destination address is valid the module will insert the data into its internal backplane command queue to transfer to the Quantum. After the request is placed in the queue the module sends the response to the master device.
5 The time period for the data to transfer to the Quantum processor will basically depend on the processor's scan rate. Depending on the processor scan rate the module could receive a command while the queue is full. In this situation the module will send a "Slave Device Busy" exception response (exception code 6) to the modbus master and the data will not be copied to the processor.
6 The module writes the data directly to the Quantum address defined by the modbus command. For the example from item 1 the data would be written starting at Quantum address 40501.

The following flow chart show the implementation of the direct pass-thru block:



The following modbus commands are accepted during direct pass-thru operation:

Function Code	Description Code	Maximum Count	Quantum Memory Area
6	Preset Single Register	1	4x (holding register)
16	Preset Multiple Registers	125	4x (holding register)

The destination address where the data will be copied at the processor is calculated as: the module will add 40001 to the destination address received.

Both ports can operate in pass-thru mode simultaneously. If the port is configured for pass-thru swap the module will swap the data bytes (high order/low order) before transferring the data to the processor.

Since the module transfers the data directly to the processor the PTQ-MCM database is not updated during pass-thru operation.

Block Pass-Thru

The following modbus functions are supported during block pass-thru operation:

Function Code	Description Code	Maximum Count	Quantum Memory Area
5*	Force Single Coil	1	0x (coil)
6	Preset Single Register	1	4x (holding register)
15*	Force Multiple Coils	800	0x (coil)
16	Preset Multiple Registers	125	4x (holding register)

*Not supported during block pass-thru with byte swap.

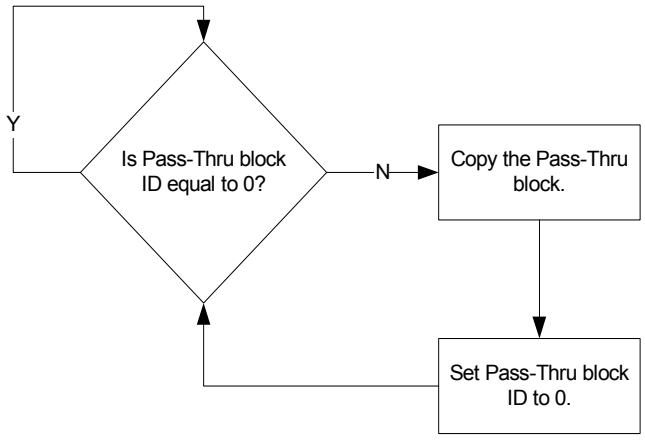
You must configure the following parameter from the configuration file during block pass-thru operation:

```
Pass Thru Address   : 1000 # Pass-Thru address
```

This parameter sets the holding register address where the pass-thru block will be copied at the processor memory. The module will add 400001 to this value, therefore for the example above the block will be copied starting at holding register address 401001.

The module writes the pass-thru block to the processor during one scan. Therefore the user should add logic in order to detect the presence of one of the block ID's listed in this section. After the block is handled the program should reset the block ID in order to use the Block ID change as a trigger for the incoming block.

The following flowchart shows the implementation of the processor logic to copy the data from the pass-thru block.



Block Pass-Thru Control Blocks

If one or more of the slave ports on the module are configured for the block pass-through mode of operation, the module will pass blocks with specific identification codes to the processor for each received write command. Any Modbus function 5, 6, 15 or 16 commands will be passed from the port to the processor. logic must handle the receipt of all Modbus write functions to the processor and to respond as expected to commands issued by the remote Modbus master device. The structure of the formatted pass-through control block is shown in the following tables:

Function 5

Offset	Description	Length
1	9958	1
2	2	1
3	Bit Address	1
4	Data	1

Function 6 and 16

Offset	Description	Length
1	9956/9957 (Floating-point)	1
2	Number of data words	1
3	Data Address	1
4 to 129	Data	125

Function 15

When the module receives a function code 15 when in pass-through mode, the module will write the data using block ID 9959 for multiple-bit data. First the bit mask is used to clear the bits to be updated. This is accomplished by ANDing the inverted mask with the existing data. Next the new data ANDed with the mask is ORed with the existing data. This protects the other bits in the INT registers from being affected.

Offset	Description	Length
1	9959	1
2	Number of Words	1
3	Word Address	1
4 to 53	Data	50
54 to 103	Mask	50

Example - Function 15

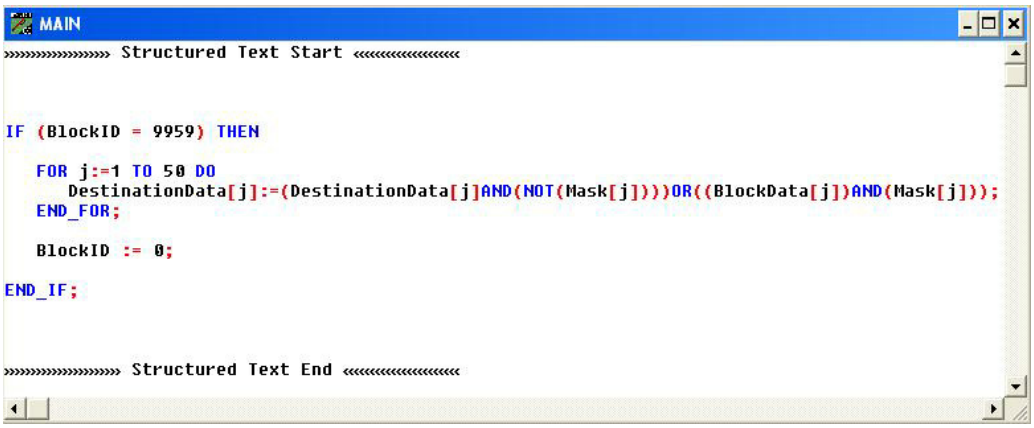
The following example shows how to use the pass-thru block functionality with modbus function 15, assuming that the parameters below were configured by the user as follows:

Parameter	Value	Description
Type	2	Port is configured as block pass-thru
Pass Thru Address	1000	The pass-thru block is copied to the processor starting at address 41001

In this example, the following variables are used:

Variable	Address	Type	Description
BlockID	401001	INT	Variable that contains the block ID. When this value is equal to 9959 the processor will trigger the logic to handle the pass-thru block.
BlockData	401004	WordArr128	This variable is an array that contains the data within the pass-thru block
Mask	401054	WordArr128	This variable is an array that contains the mask within the pass-thru block
DestinationData	Any	WordArr128	This variable is an array where the final data will be copied to. Only the masked bits will be copied to this variable.

The following is a sample program in structured text that shows how the data could be transferred to the processor using the mask information.



8.3 Status Data Definition

This section contains a description of the members present in the status block. This data is transferred from the module to the processor as part of each read block.

8.3.1 Status Data Block Structure

Offset	Content	Description	Length (Words)
0	Program Scan Count	This value is incremented each time a complete program cycle occurs in the module.	1
1	Product Code	These two registers contain the product code of "MCM"	2
3	Product Version	These two registers contain the product version for the current running software.	2
5	Operating System	These two registers contain the month and year values for the program operating system.	2

Offset	Content	Description	Length (Words)
7	Run Number	These two registers contain the run number value for the currently running software.	2
9	Port 1 Command List Requests	This field contains the number of requests made from this port to slave devices on the network.	1
10	Port 1 Command List Response	This field contains the number of slave response messages received on the port.	1
11	Port 1 Command List Errors	This field contains the number of command errors processed on the port. These errors could be due to a bad response or command.	1
12	Port 1 Requests	This field contains the total number of messages sent out of the port.	1
13	Port 1 Responses	This field contains the total number of messages received on the port.	1
14	Port 1 Errors Sent	This field contains the total number of message errors sent out of the port.	1
15	Port 1 Errors Received	This field contains the total number of message errors received on the port.	1
16	Port 2 Command List Requests	This field contains the number of requests made from this port to slave devices on the network.	1
17	Port 2 Command List Response	This field contains the number of slave response messages received on the port.	1
18	Port 2 Command List Errors	This field contains the number of command errors processed on the port. These errors could be due to a bad response or command.	1
19	Port 2 Requests	This field contains the total number of messages sent out the port.	1
20	Port 2 Responses	This field contains the total number of messages received on the port.	1
21	Port 2 Errors Sent	This field contains the total number of message errors sent out of the port.	1
22	Port 2 Errors Received	This field contains the total number of message errors received on the port.	1
23	Read Block Count	This field contains the total number of read blocks transferred from the module to the processor.	1
24	Write Block Count	This field contains the total number of write blocks transferred from the processor to the module.	1
25	Parse Block Count	This field contains the total number of blocks successfully parsed that were received from the processor.	1
26	Command Event Block Count	This field contains the total number of command event blocks received from the processor.	1
27	Command Block Count	This field contains the total number of command blocks received from the processor.	1
28	Error Block Count	This field contains the total number of block errors recognized by the module.	1

Offset	Content	Description	Length (Words)
29	Pass Thru Received	Pass thru request received on both ports.	1
30	Port 1 Current Error	For a slave port, this field contains the value of the current error code returned. For a master port, this field contains the index of the currently executing command.	1
31	Port 1 Last Error	For a slave port, this field contains the value of the last error code returned. For a master port, this field contains the index of the command with an error.	1
32	Port 2 Current Error	For a slave port, this field contains the value of the current error code returned. For a master port, this field contains the index of the currently executing command.	1
33	Port 2 Last Error	For a slave port, this field contains the value of the last error code returned. For a master port, this field contains the index of the command with an error.	1

8.4 Configuration Data

This section contains listings of the PTQ-MCM module's database that are related to the module's configuration. This data is available to any node on the network and is read from the Quantum processor when the module first initializes.

Refer to this section for information about the configuration file parameters.

8.4.1 Port 1 Setup

Database Address	Content
5010	Enable
5011	Pass-Thru
5012	Type
5013	Float Flag
5014	Float Start
5015	Float Offset
5016	Protocol
5017	Baud Rate
5018	Parity
5019	Data Bits
5020	Stop Bits
5021	RTS On
5022	RTS Off
5023	Minimum Response Time
5024	Use CTS Line
5025	Slave ID
5026	Bit in Offset
5027	Word in Offset
5028	Out in Offset

Database Address	Content
5029	Holding Reg Offset
5030	Command Count
5031	Minimum Command Delay
5032	Command Error Pointer
5033	Response Timeout
5034	Retry Count
5035	Error Delay Counter

8.4.2 Port 2 Setup

Database Address	Content
5040	Enable
5041	Passthru
5042	Type
5043	Float Flag
5044	Float Start
5045	Float Offset
5046	Protocol
5047	Baud Rate
5048	Parity
5049	Data Bits
5050	Stop Bits
5051	RTS On
5052	RTS Off
5053	Minimum Response Time
5054	Use CTS Line
5055	Slave ID
5056	Bit in Offset
5057	Word in Offset
5058	Out in Offset
5059	Holding Reg Offset
5060	Command Count
5061	Minimum Command Delay
5062	Command Error Pointer
5063	Response Timeout
5064	Retry Count
5065	Error Delay Counter

8.4.3 Port 1 Commands

Database Address	Content
5070	Command #1
5078	Command #2
-	-
5862	Command #100

Each of these commands uses eight registers (Enable, Internal Address, Poll Interval, Reg Count, Swap Code, Node Address, Modbus Func, MB Address in Device).

8.4.4 Port 2 Commands

Database Address	Content
5870	Command #1
5877	Command #2
-	-
6662	Command #100

Each of these commands uses eight registers (Enable, Internal Address, Poll Interval, Reg Count, Swap Code, Node Address, Modbus Func, MB Address in Device).

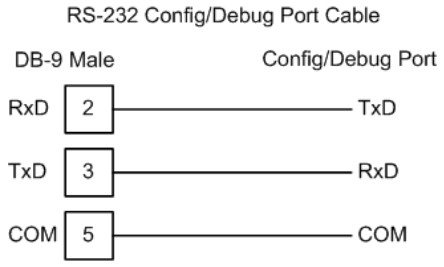
8.5 Port Specific Information

8.5.1 Serial Port Details

The ProTalk module contains three serial ports. The top port is the Configuration/Debug port and is used for performing tasks such as uploading and downloading configuration files, viewing module status, running diagnostics, and troubleshooting. This port is a standard serial port set up for RS-232 communications.

8.5.2 RS-232 Configuration/Debug Port

This port is physically a DB-9 connection. This port permits a PC based terminal emulation program to view configuration and status data in the module and to control the module. The cable for communications on this port is shown below:



8.6 Modbus Protocol Specification

8.6.1 Read Output Status (Function Code 01)

Query

This function allows the user to obtain the ON/OFF status of logic coils used to control discrete outputs from the addressed slave only. Broadcast mode is not supported with this function code. In addition to the slave address and function fields, the message requires that the information field contain the initial coil address to be read (Starting Address) and the number of locations that will be interrogated to obtain status data.

The addressing allows up to 2000 coils to be obtained at each request; however, the specific slave device may have restrictions that lower the maximum quantity. The coils are numbered from zero; (coil number 1 = zero, coil number 2 = one, coil number 3 = two, etc.).

The following table is a sample read output status request to read coils 0020 to 0056 from slave device number 11.

Adr	Func	Data Start Pt Hi	Data Start Pt Lo	Data # Of Pts Ho	Data # Of Pts Lo	Error Check Field
11	01	00	13	00	25	CRC

Response

An example response to Read Output Status is as shown in Figure C2. The data is packed one bit for each coil. The response includes the slave address, function code, quantity of data characters, the data characters, and error checking. Data will be packed with one bit for each coil (1 = ON, 0 = OFF). The low order bit of the first character contains the addressed coil, and the remainder follow. For coil quantities that are not even multiples of eight, the last characters will be filled in with zeros at high order end. The quantity of data characters is always specified as quantity of RTU characters, that is, the number is the same whether RTU or ASCII is used.

Because the slave interface device is serviced at the end of a controller's scan, data will reflect coil status at the end of the scan. Some slaves will limit the quantity of coils provided each scan; thus, for large coil quantities, multiple PC transactions must be made using coil status from sequential scans.

Adr	Func	Byte Count	Data Coil Status 20 to 27	Data Coil Status 28 to 35	Data Coil Status 36 to 43	Data Coil Status 44 to 51	Data Coil Status 52 to 56	Error Check Field
11	01	05	CD	6B	B2	OE	1B	CRC

The status of coils 20 to 27 is shown as CD(HEX) = 1100 1101 (Binary). Reading left to right, this shows that coils 27, 26, 23, 22, and 20 are all on. The other coil data bytes are decoded similarly. Due to the quantity of coil statuses requested, the last data field, which is shown 1B (HEX) = 0001 1011 (Binary), contains the status of only 5 coils (52 to 56) instead of 8 coils. The 3 left most bits are provided as zeros to fill the 8-bit format.

8.6.2 Read Input Status (Function Code 02)

Query

This function allows the user to obtain the ON/OFF status of discrete inputs in the addressed slave PC Broadcast mode is not supported with this function code. In addition to the slave address and function fields, the message requires that the information field contain the initial input address to be read (Starting Address) and the number of locations that will be interrogated to obtain status data.

The addressing allows up to 2000 inputs to be obtained at each request; however, the specific slave device may have restrictions that lower the maximum quantity. The inputs are numbered form zero; (input 10001 = zero, input 10002 = one, input 10003 = two, etc., for a 584).

The following table is a sample read input status request to read inputs 10197 to 10218 from slave number 11.

Adr	Func	Data Start Pt Hi	Data Start Pt Lo	Data #of Pts Hi	Data #of Pts Lo	Error Check Field
11	02	00	C4	00	16	CRC

Response

An example response to Read input status is as shown in Figure C4. The data is packed one bit for each input. The response includes the slave address, function code, quantity of data characters, the data characters, and error checking. Data will be packed with one bit for each input (1=ON, 0=OFF). The lower order bit of the first character contains the addressed input, and the remainder follow. For input quantities that are not even multiples of eight, the last characters will be filled in with zeros at high order end. The quantity of data characters is always specified as a quantity of RTU characters, that is, the number is the same whether RTU or ASCII is used.

Because the slave interface device is serviced at the end of a controller's scan, data will reflect input status at the end of the scan. Some slaves will limit the quantity of inputs provided each scan; thus, for large coil quantities, multiple PC transactions must be made using coil status for sequential scans.

Adr	Func	Byte Count	Data Discrete Input 10197 to 10204	Data Discrete Input 10205 to 10212	Data Discrete Input 10213 to 10218	Error Check Field
11	02	03	AC	DB	35	CRC

The status of inputs 10197 to 10204 is shown as AC (HEX) = 10101 1100 (binary). Reading left to right, this show that inputs 10204, 10202, and 10199 are all on. The other input data bytes are decoded similar.

Due to the quantity of input statuses requested, the last data field which is shown as 35 HEX = 0011 0101 (binary) contains the status of only 6 inputs (10213 to 102180) instead of 8 inputs. The two left-most bits are provided as zeros to fill the 8-bit format.

8.6.3 Read Holding Registers (Function Code 03)

Query

Read holding registers (03) allows the user to obtain the binary contents of holding registers 4xxxx in the addressed slave. The registers can store the numerical values of associated timers and counters which can be driven to external devices. The addressing allows up to 125 registers to obtained at each request; however, the specific slave device may have restriction that lower this maximum quantity. The registers are numbered form zero (40001 = zero, 40002 = one, etc.). The broadcast mode is not allowed.

The example below reads registers 40108 through 40110 from slave 584 number 11.

Adr	Func	Data Start Reg Hi	Data Start Reg Lo	Data #of Regs Hi	Data #of Regs Lo	Error Check Field
11	03	00	6B	00	03	CRC

Response

The addressed slave responds with its address and the function code, followed by the information field. The information field contains 1 byte describing the quantity of data bytes to be returned. The contents of the registers requested (DATA) are two bytes each, with the binary content right justified within each pair of characters. The first byte includes the high order bits and the second, the low order bits.

Because the slave interface device is normally serviced at the end of the controller's scan, the data will reflect the register content at the end of the scan. Some slaves will limit the quantity of register content provided each scan; thus for large register quantities, multiple transmissions will be made using register content from sequential scans.

In the example below, the registers 40108 to 40110 have the decimal contents 555, 0, and 100 respectively.

Adr	Func	ByteCnt	Hi Data	Lo Data	Hi Data	Lo Data	Hi Data	Lo Data	Error Check Field
11	03	06	02	2B	00	00	00	64	CRC

8.6.4 Read Input Registers (Function Code 04)

Query

Function code 04 obtains the contents of the controller's input registers at addresses 3xxxx. These locations receive their values from devices connected to the I/O structure and can only be referenced, not altered from within the controller, The addressing allows up to 125 registers to be obtained at each request; however, the specific slave device may have restrictions that lower this maximum quantity. The registers are numbered for zero (30001 = zero, 30002 = one, etc.). Broadcast mode is not allowed.

The example below requests the contents of register 3009 in slave number 11.

Adr	Func	Data Start Reg Hi	Data Start Reg Lo	Data #of Regs Hi	Data #of Regs Lo	Error Check Field
11	04	00	08	00	01	CRC

Response

The addressed slave responds with its address and the function code followed by the information field. The information field contains 1 byte describing the quantity of data bytes to be returned. The contents of the registers requested (DATA) are 2 bytes each, with the binary content right justified within each pair of characters. The first byte includes the high order bits and the second, the low order bits.

Because the slave interface is normally serviced at the end of the controller's scan, the data will reflect the register content at the end of the scan. Each PC will limit the quantity of register contents provided each scan; thus for large register quantities, multiple PC scans will be required, and the data provided will be form sequential scans.

In the example below the register 3009 contains the decimal value 0.

Adr	Func	Byte Count	Data Input Reg Hi	Data Input Reg Lo	Error Check Field
11	04	02	00	00	E9

8.6.5 Force Single Coil (Function Code 05)

Query

This message forces a single coil either ON or OFF. Any coil that exists within the controller can be forced to either state (ON or OFF). However, because the controller is actively scanning, unless the coil is disabled, the controller can also alter the state of the coil. Coils are numbered from zero (coil 0001 = zero, coil 0002 = one, etc.). The data value 65,280 (FF00 HEX) will set the coil ON and the value zero will turn it OFF; all other values are illegal and will not affect that coil.

The use of slave address 00 (Broadcast Mode) will force all attached slaves to modify the desired coil.

Note: Functions 5, 6, 15, and 16 are the only messages that will be recognized as valid for broadcast.

The example below is a request to slave number 11 to turn ON coil 0173.

Adr	Func	Data Coil # Hi	Data Coil # Lo	Data On/off Ind	Data	Error Check Field
11	05	00	AC	FF	00	CRC

Response

The normal response to the Command Request is to re-transmit the message as received after the coil state has be altered.

Adr	Func	Data Coil # Hi	Data Coil # Lo	Data On/ Off	Data	Error Check Field
11	05	00	AC	FF	00	CRC

The forcing of a coil via MODBUS function 5 will be accomplished regardless of whether the addressed coil is disabled or not. (*In ProSoft products,* the coil *is only affected if the necessary ladder logic is implemented).*

Note: The Modbus protocol does not include standard functions for testing or changing the DISABLE state of discrete inputs or outputs. Where applicable, this may be accomplished via device specific Program commands. (*In ProSoft products, this is only accomplished through ladder logic programming).*

Coils that are reprogrammed in the controller logic program are not automatically cleared upon power up. Thus, if such a coil is set ON by function Code 5 and (even months later), an output is connected to that coil, the output will be "hot".

8.6.6 Preset Single Register (Function Code 06)

Query

Function (06) allows the user to modify the contents of a holding register. Any holding register that exists within the controller can have its contents changed by this message. However, because the controller is actively scanning, it also can alter the content of any holding register at any time. The values are provided in binary up to the maximum capacity of the controller unused high order bits must be set to zero. When used with slave address zero (Broadcast mode) all slave controllers will load the specified register with the contents specified.

NOTE Functions 5, 6, 15 and 16 are the only messages that will be recognized as valid for broadcast.

Adr	Func	Data Start Reg Hi	Data Start Reg Lo	Data #of Regs Hi	Data #of Regs Lo	Error Check Field
11	06	00	01	00	03	CRC

Response

The response to a preset single register request is to re-transmit the query message after the register has been altered.

Adr	Func	Data Reg Hi	Data Reg Lo	Data Input Reg Hi	Data Input Reg Lo	Error Check Field
11	06	00	01	00	03	CRC

8.6.7 Diagnostics (Function Code 08)

MODBUS function code 08 provides a series of tests for checking the communication system between a client (Master) device and a server (Slave), or for checking various internal error conditions within a server.

The function uses a two-byte sub-function code field in the query to define the type of test to be performed. The server echoes both the function code and sub-function code in a normal response. Some of the diagnostics cause data to be returned from the remote device in the data field of a normal response.

In general, issuing a diagnostic function to a remote device does not affect the running of the user program in the remote device. User logic, like discrete and registers, is not accessed by the diagnostics. Certain functions can optionally reset error counters in the remote device.

A server device can, however, be forced into 'Listen Only Mode' in which it will monitor the messages on the communications system but not respond to them. This can affect the outcome of your application program if it depends upon any further exchange of data with the remote device. Generally, the mode is forced to remove a malfunctioning remote device from the communications system.

The following diagnostic functions are dedicated to serial line devices.

The normal response to the Return Query Data request is to loopback the same data. The function code and sub-function codes are also echoed.

Request		
Function code	1 Byte	0x08
Sub-function	2 Bytes	
Data	N x 2 Bytes	

Response		
Function code	1 Byte	0x08
Sub-function	2 Bytes	
Data	N x 2 Bytes	

Error		
Error code	1 Byte	0x88
Exception code	1 Byte	01 or 03 or 04

Sub-function codes supported by the serial line devices

Here the list of sub-function codes supported by the serial line devices. Each sub-function code is then listed with an example of the data field contents that would apply for that diagnostic.

Sub-functions supported by the PTQ-MCM module are highlighted in **BOLD**.

Sub-function code		**Name**
Hex	**Dec**	
00	00	Return Query Data
01	01	Restart Communications Option
02	02	Return Diagnostic Register
03	03	Change ASCII Input Delimiter
04	04	Force Listen Only Mode
05..09	05..09	RESERVED
0A	10	Clear Counters and Diagnostic Register
0B	11	Return Bus Message Count
0C	12	Return Bus Communication Error Count
0D	13	Return Bus Exception Error Count
0E	14	Return Slave Message Count
0F	15	Return Slave No Response Count
10	16	Return Slave NAK Count
11	17	Return Slave Busy Count
12	18	Return Bus Character Overrun Count
13	19	RESERVED
14	20	Clear Overrun Counter and Flag
N.A.	21..65535	Reserved

00 Return Query Data

The data passed in the request data field is to be returned (looped back) in the response. The entire response message should be identical to the request.

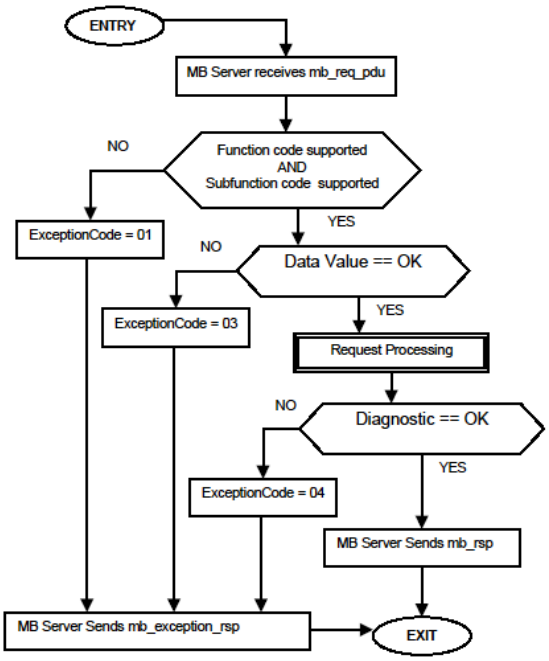
Sub-function	Data Field (Request)	Data Field (Response)
00 00	Any	Echo Request Data

Example and state diagram

Here is an example of a request to remote device to Return Query Data. This uses a sub-function code of zero (00 00 hex in the two-byte field). The data to be returned is sent in the two-byte data field (A5 37 hex).

Request		Response	
Field Name	**(Hex)**	**Field Name**	**(Hex)**
Function	08	Function	08
Sub-function Hi	00	Sub-function Hi	00
Sub-function Lo	00	Sub-function Lo	00
Data Hi	A5	Data Hi	A5
Data Lo	37	Data Lo	27

The data fields in responses to other kinds of queries could contain error counts or other data requested by the sub-function code.



8.6.8 Force Multiple Coils (Function Code 15)

Query

This message forces each coil in a consecutive block of coils to a desired ON or OFF state. Any coil that exists within the controller can be forced to either state (ON or OFF). However, because the controller is actively scanning, unless the coils are disabled, the controller can also alter the state of the coil. Coils are numbered from zero (coil 00001 = zero, coil 00002 = one, etc.). The desired status of each coil is packed in the data field, one bit for each coil (1= ON, 0= OFF). The use of slave address 0 (Broadcast Mode) will force all attached slaves to modify the desired coils.

Note: Functions 5, 6, 15, and 16 are the only messages (other than Loopback Diagnostic Test) that will be recognized as valid for broadcast.

The following example forces 10 coils starting at address 20 (13 HEX). The two data fields, CD =1100 and 00 = 0000 000, indicate that coils 27, 26, 23, 22, and 20 are to be forced on.

Adr	Func	Hi Add	Lo Add	Quantity	Byte Cnt	Data Coil Status 20 to 27	Data Coil Status 28 to 29	Error Check Field	
11	0F	00	13	00	0A	02	CD	00	CRC

Response

The normal response will be an echo of the slave address, function code, starting address, and quantity of coils forced.

Adr	Func	Hi Addr	Lo Addr	Quantity	Error Check Field	
11	0F	00	13	00	0A	CRC

The writing of coils via Modbus function 15 will be accomplished regardless of whether the addressed coils are disabled or not.

Coils that are unprogrammed in the controller logic program are not automatically cleared upon power up. Thus, if such a coil is set ON by function code 15 and (even months later) an output is connected to that coil, the output will be hot.

8.6.9 Preset Multiple Registers (Function Code 16)

Query

Holding registers existing within the controller can have their contents changed by this message (a maximum of 60 registers). However, because the controller is actively scanning, it also can alter the content of any holding register at any time. the values are provided in binary up to the maximum capacity of the controller (16-bit for the 184/384 and 584); unused high order bits must be set to zero. When specified registers with contents specified.

Note: Function codes 5, 6, 15, and 16 are the only messages that will be recognized as valid for broadcast.

Adr	Func	Hi Add	Lo Add	Quantity		Byte Cnt	Hi Data	Lo Data	Hi Data	Lo Data	Error Check Field
11	10	00	87	00	02	04	00	0A	01	02	CRC

Response

The normal response to a function 16 query is to echo the address, function code, starting address and number of registers to be loaded.

Adr	Func	Hi Addr	Lo Addr	Quantity		Error Check Field
11	10	00	87	00	02	56

8.6.10 MODBUS Exception Responses

When a client device sends a request to a server device it expects a normal response. One of four possible events can occur from the master's query:

- If the server device receives the request without a communication error, and can handle the query normally, it returns a normal response.
- If the server does not receive the request due to a communication error, no response is returned. The client program will eventually process a timeout condition for the request.
- If the server receives the request, but detects a communication error (parity, LRC, CRC, ...), no response is returned. The client program will eventually process a timeout condition for the request.
- If the server receives the request without a communication error, but cannot handle it (for example, if the request is to read a non-existent output or register), the server will return an exception response informing the client of the nature of the error.

The exception response message has two fields that differentiate it from a normal response:

Function Code Field: In a normal response, the server echoes the function code of the original request in the function code field of the response. All function codes have a most-significant bit (MSB) of 0 (their values are all below 80 hexadecimal). In an exception response, the server sets the MSB of the function code to 1. This makes the function code value in an exception response exactly 80 hexadecimal higher than the value would be for a normal response.

With the function code's MSB set, the client's application program can recognize the exception response and can examine the data field for the exception code.

Data Field: In a normal response, the server may return data or statistics in the data field (any information that was requested in the request). In an exception response, the server returns an exception code in the data field. This defines the server condition that caused the exception.

The following table shows an example of a client request and server exception response.

Request		**Response**	
Field Name	**(Hex)**	**Field Name**	**(Hex)**
Function	01	Function	81
Starting Address Hi	04	Exception Code	02
Starting Address Lo	A1		
Quantity of Outputs Hi	00		
Quantity of Outputs Lo	01		

In this example, the client addresses a request to server device. The function code (01) is for a Read Output Status operation. It requests the status of the output at address 1245 (04A1 hex). Note that only that one output is to be read, as specified by the number of outputs field (0001).

If the output address is non-existent in the server device, the server will return the exception response with the exception code shown (02). This specifies an illegal data address for the slave.

MODBUS Exception Codes

Code	Name	Meaning
01	Illegal Function	The function code received in the query is not an allowable action for the server (or slave). This may be because the function code is only applicable to newer devices, and was not implemented in the unit selected. It could also indicate that the server (or slave) is in the wrong state to process a request of this type, for example because it is unconfigured and is being asked to return register values.
02	Illegal Data Address	The data address received in the query is not an allowable address for the server (or slave). More specifically, the combination of reference number and transfer length is invalid. For a controller with 100 registers, a request with offset 96 and length 4 would succeed, a request with offset 96 and length 5 will generate exception 02.
03	Illegal Data Value	A value contained in the query data field is not an allowable value for server (or slave). This indicates a fault in the structure of the remainder of a complex request, such as that the implied length is incorrect. It specifically does not mean that a data item submitted for storage in a register has a value outside the expectation of the application program, because the Modbus protocol is unaware of the significance of any particular value of any particular register.
04	Slave Device Failure	An unrecoverable error occurred while the server (or slave) was attempting to perform the requested action.
05	Acknowledge	Specialized use in conjunction with programming commands. The server (or slave) has accepted the request and is processing it, but a long duration of time will be required to do so. This response is returned to prevent a timeout error from occurring in the client (or master). The client (or master) can next issue a poll program complete message to determine if processing is completed.
06	Slave Device Busy	Specialized use in conjunction with programming commands. The server (or slave) is engaged in processing a long-duration program command. The client (or master) should retransmit the message later when the server (or slave) is free.

Code	Name	Meaning
08	Memory Parity Error	Specialized use in conjunction with function codes 20 and 21 and reference type 6, to indicate that the extended file area failed to pass a consistency check. The server (or slave) attempted to read record file, but detected a parity error in the memory. The client (or master) can retry the request, but service may be required on the server (or slave) device.
0a	Gateway Path Unavailable	Specialized use in conjunction with gateways, indicates that the gateway was unable to allocate an internal communication path from the input port to the output port for processing the request. Usually means that the gateway is misconfigured or overloaded.
0b	Gateway Target Device Failed To Respond	Specialized use in conjunction with gateways, indicates that no response was obtained from the target device. Usually means that the device is not present on the network.

8.7 Frequently Asked Questions

8.7.1 What kind of data transfer rates can I expect between the PLC and the module?

Data transfer rates between the PLC and the module depend on a number of variables, among them the number of words being transferred per command, the amount of other network traffic at the time data is being transferred, and overall processor scan times.

8.7.2 Is a .MDC available for configuration of the Module?

Yes. The CDROM that ships with the module should have a version for both Concept 2.5 and 2.6 in the ProTalk directory.

8.7.3 Does the module work in a remote rack?

he module is designed to be located in the chassis with the PLC and will not operate in a remote chassis. If your application requires remote placement of the communication device you should investigate the other members of the ProLinx family such as the 4202-MNET-DFCM. (if you require DF1 connectivity for instance although many others are available) This module for example would allow you to communicate with DF1 devices and allow you to map the contents of its memory using Modbus TCP/IP.

8.7.4 Can I use the module in a hot backup system?

Support for Hot Backup is not currently implemented in the module. We are currently investigating the addition of this functionality but until this development can be finalized, it may be possible to use one of the 4000 series of ProSoft communication products. Please call our technical support technicians when considering this application.

9 Support, Service & Warranty

In This Chapter

- How to Contact Us: Sales and Support 128
- Return Material Authorization (RMA) Policies and Conditions..... 128
- Procedures for Return of Units Under Warranty 129
- Procedures for Return of Units Out of Warranty 130
- LIMITED WARRANTY 131

Be sure and read the full Warranty that can be found on our web site at www.prosoft-technology.com for details and other terms and conditions. The content in this summary is subject to change without notice. The content is current at date of publication.

ProSoft Technology, Inc. strives to provide meaningful support to its customers. Should any questions or problems arise, please feel free to contact us at:

Internet	Web Site: http://www.prosoft-technology.com/support
	E-mail address: support@prosoft-technology.com

Those of us at ProSoft Technology, Inc. want to provide the best and quickest support possible, so before calling please have the following information available. You may wish to fax this information to us prior to calling.

1 Product Version Number
2 System architecture
3 Network details

In the case of hardware, we will also need the following information:

1 Module configuration and contents of file
2 Module Operation
3 Configuration/Debug status information
4 LED patterns
5 Information about the processor and user data files as viewed through the development software and LED patterns on the processor
6 Details about the networked devices interfaced, if any

For technical support calls within the United States, an after-hours answering system allows pager access to one of our qualified technical and/or application support engineers at any time to answer your questions.

9.1 How to Contact Us: Sales and Support

All ProSoft Technology Products are backed with full technical support. Contact our worldwide Technical Support team and Customer Service representatives directly by phone or email:

USA / Latin America (excluding Brasil) (Office in California)

+1(661) 716-5100
+1(661) 716-5101 (Fax)
1675 Chester Avenue, 4th Floor
Bakersfield, California 93301
U.S.A.
+1.661.716.5100, support@prosoft-technology.com
Languages spoken include: English, Spanish

Asia Pacific Sales (office in Malaysia)

+603.7724.2080
+603.7724.2090 (Fax)
C210, Damansara Intan,
1 Jalan SS20/27, 47400 Petaling Jaya
Selangor, Malaysia
+603.7724.2080, asiapc@prosoft-technology.com
Languages spoken include: Chinese, Japanese, English

Asia Pacific Support (office in China)

+86.21.64518356 x 8011
+86.21.64756957 (Fax)
4/F, No. 16 Hongcao Road
Shanghai, China 200233
China
+86.21.64518356 x 8011, zhang@prosoft-technology.com
Languages spoken include: Chinese, English

Europe / Middle East / Africa (office in Toulouse, France)

+33 (0) 5.34.36.87.20
+33 (0) 5.61.78.40.52 (Fax)
Zone d'activité de Font Grasse
17, rue des Briquetiers
F-31700 Blagnac
France
+33 (0) 5.34.36.87.20. support.emea@prosoft-technology.com
Languages spoken include: French, English

Brasil (office in Sao Paulo)

+55-11-5084-5178
+55-11-5083-3776 (Fax)
Rua Vergueiro, 2949 - sala 182 - Edifício Vergueiro Work Center
Vila Mariana - São Paulo
Cep: 04101-300 - Brasil
+55-11-5084-5178, eduardo@prosoft-technology.com
Languages spoken include: Portuguese, English

9.2 Return Material Authorization (RMA) Policies and Conditions

The following RMA Policies and Conditions apply to any returned product. These RMA Policies are subject to change by ProSoft without notice. For warranty information, see Section C below entitled "Limited Warranty". In the event of any inconsistency between the RMA Policies and the Warranty, the Warranty shall govern.

9.2.1 All Product Returns

1 In order to return a Product for repair, exchange or otherwise, the Customer must obtain a Returned Material Authorization (RMA) number from ProSoft and comply with ProSoft shipping instructions.
2 In the event that the Customer experiences a problem with the Product for any reason, Customer should contact ProSoft Technical Support at one of the telephone numbers listed above in Section A. A Technical Support Engineer will request several tests in an attempt to isolate the problem. If after these tests are completed, the Product is found to be the source of the problem, ProSoft will issue an RMA.
3 All returned Products must be shipped freight prepaid, in the original shipping container or equivalent, to the location specified by ProSoft, and be accompanied by proof of purchase. The RMA number is to be prominently marked on the outside of the shipping box. Customer agrees to insure the Product or assume the risk of loss or damage in transit. Products shipped to ProSoft without an RMA number will be returned to the Customer, freight collect. Contact ProSoft Technical Support for further information.
4 Out of warranty returns are not allowed on RadioLinx accessories such as antennas, cables, and brackets.

The following policy applies for Non-Warranty Credit Returns:

A 10% Restocking Fee if Factory Seal is *not* broken
B 20% Restocking Fee if Factory Seal is broken

ProSoft retains the right, in its absolute and sole discretion, to reject any non-warranty returns for credit if the return is not requested within three (3) months after shipment of the Product to Customer, if the Customer fails to comply with ProSoft's shipping instructions, or if the Customer fails to return the Product to ProSoft within six (6) months after Product was originally shipped.

9.3 Procedures for Return of Units Under Warranty

1 A Technical Support Engineer must pre-approve all product returns.
2 Module is repaired or replaced after a Return Material Authorization Number is entered and a replacement order is generated.
3 Credit for the warranted item is issued within 10 business days after receipt of product and evaluation of the defect has been performed by ProSoft. The credit will only be issued provided the product is returned with a valid Return Material Authorization Number and in accordance with ProSoft's shipping instructions.

a) If no defect is found, a credit is issued.

b) If a defect is found and is determined to be customer generated or if the defect is otherwise not covered by ProSoft's Warranty, or if the module is not repairable, a credit is not issued and payment of the replacement module is due.

9.4 Procedures for Return of Units Out of Warranty

1 Customer sends unit in for evaluation.

2 If no defect is found, Customer will be charged the equivalent of US $100 plus shipping, duties and taxes that may apply. A new Purchase Order will be required for this evaluation fee.

If the unit is repaired the charge to the Customer will be 30%* of the list price plus any shipping, duties and taxes that may apply. A new Purchase Order will be required for a product repair.

3 For an immediate exchange, a new module may be purchased and sent to Customer while repair work is being performed. Credit for purchase of the new module will be issued when the new module is returned in accordance with ProSoft's shipping instructions and subject to ProSoft's policy on non-warranty returns. This is in addition to charges for repair of the old module and any associated charges to Customer.

4 If, upon contacting ProSoft Customer Service, the Customer is informed that unit is believed to be unrepairable, the Customer may choose to send unit in for evaluation to determine if the repair can be made. Customer will pay shipping, duties and taxes that may apply. If unit cannot be repaired, the Customer may purchase a new unit.

9.4.1 Un-repairable Units

- 3150-All
- 3750
- 3600-All
- 3700
- 3170-All
- 3250
- 1560 can be repaired, if defect is the power supply
- 1550 can be repaired, if defect is the power supply
- 3350
- 3300
- 1500-All

*** 30% of list price is an estimated repair cost only. The actual cost of repairs will be determined when the module is received by ProSoft and evaluated for needed repairs.**

Purchasing Warranty Extension

As detailed below in ProSoft's Warranty, the standard Warranty Period is one year (or in the case of RadioLinx modules, three years) from the date of delivery. The Warranty Period may be extended for an additional charge, as follows:

- Additional 1 year = 10% of list price
- Additional 2 years = 20% of list price
- Additional 3 years = 30% of list price

9.5 LIMITED WARRANTY

This Limited Warranty ("Warranty") governs all sales of hardware, software and other products (collectively, "Product") manufactured and/or offered for sale by ProSoft, and all related services provided by ProSoft, including maintenance, repair, warranty exchange, and service programs (collectively, "Services"). By purchasing or using the Product or Services, the individual or entity purchasing or using the Product or Services ("Customer") agrees to all of the terms and provisions (collectively, the "Terms") of this Limited Warranty. All sales of software or other intellectual property are, in addition, subject to any license agreement accompanying such software or other intellectual property.

9.5.1 What Is Covered By This Warranty

A *Warranty On New Products*: ProSoft warrants, to the original purchaser only, that the Product that is the subject of the sale will (1) conform to and perform in accordance with published specifications prepared, approved, and issued by ProSoft, and (2) will be free from defects in material or workmanship; provided these warranties only cover Product that is sold as new. This Warranty expires one year (or in the case of RadioLinx modules, three years) from the date of shipment (the "Warranty Period"). If the Customer discovers within the Warranty Period a failure of the Product to conform to specifications, or a defect in material or workmanship of the Product, the Customer must promptly notify ProSoft by fax, email or telephone. In no event may that notification be received by ProSoft later than 15 months (or in the case of RadioLinx modules, 39 months) from the date of delivery. Within a reasonable time after notification, ProSoft will correct any failure of the Product to conform to specifications or any defect in material or workmanship of the Product, with either new or used replacement parts. Such repair, including both parts and labor, will be performed at ProSoft's expense. All warranty service will be performed at service centers designated by ProSoft. If ProSoft is unable to repair the Product to conform to this Warranty after a reasonable number of attempts, ProSoft will provide, at its option, one of the following: a replacement product, a full refund of the purchase price or a credit in the amount of the purchase price. All replaced product and parts become the property of ProSoft. These remedies are the Customer's only remedies for breach of warranty.

B *Warranty On Services*: Material and labor used by ProSoft to repair a verified malfunction or defect are warranted on the terms specified above for new Product, provided said warranty will be for the period remaining on the original new equipment warranty or, if the original warranty is no longer in effect, for a period of 90 days from the date of repair.

C The Warranty Period for RadioLinx accessories (such as antennas, cables, brackets, etc.) are the same as for RadioLinx modules, that is, three years from the date of shipment.

9.5.2 What Is Not Covered By This Warranty

A ProSoft makes no representation or warranty, expressed or implied, that the operation of software purchased from ProSoft will be uninterrupted or error free or that the functions contained in the software will meet or satisfy the purchaser's intended use or requirements; the Customer assumes complete responsibility for decisions made or actions taken based on information obtained using ProSoft software.

B With the exception of RadioLinx accessories referenced in paragraph 1(c) this Warranty does not cover any product, components, or parts not manufactured by ProSoft.

C This Warranty also does not cover the failure of the Product to perform specified functions, or any other non-conformance, defects, losses or damages caused by or attributable to any of the following: (i) shipping; (ii) improper installation or other failure of Customer to adhere to ProSoft's specifications or instructions; (iii) unauthorized repair or maintenance; (iv) attachments, equipment, options, parts, software, or user-created programming (including, but not limited to, programs developed with any IEC 61131-3 programming languages, or "C") not furnished by ProSoft; (v) use of the Product for purposes other than those for which it was designed; (vi) any other abuse, misapplication, neglect or misuse by the Customer; (vii) accident, improper testing or causes external to the Product such as, but not limited to, exposure to extremes of temperature or humidity, power failure or power surges outside of the limits indicated on the product specifications; or (viii) disasters such as fire, flood, earthquake, wind or lightning.

D The information in this Agreement is subject to change without notice. ProSoft shall not be liable for technical or editorial errors or omissions made herein; nor for incidental or consequential damages resulting from the furnishing, performance or use of this material. The user guides included with your original product purchased by you from ProSoft, contains information protected by copyright. No part of the guide may be duplicated or reproduced in any form without prior written consent from ProSoft.

9.5.3 DISCLAIMER REGARDING HIGH RISK ACTIVITIES

PRODUCT MANUFACTURED OR SUPPLIED BY PROSOFT IS NOT FAULT TOLERANT AND IS NOT DESIGNED, MANUFACTURED OR INTENDED FOR USE IN HAZARDOUS ENVIRONMENTS REQUIRING FAIL-SAFE PERFORMANCE (INCLUDING, WITHOUT LIMITATION, THE OPERATION OF NUCLEAR FACILITIES, AIRCRAFT NAVIGATION OF COMMUNICATION SYSTEMS, AIR TRAFFIC CONTROL, DIRECT LIFE SUPPORT MACHINES OR WEAPONS SYSTEMS), IN WHICH THE FAILURE OF THE PRODUCT COULD LEAD DIRECTLY OR INDIRECTLY TO DEATH, PERSONAL INJURY, OR SEVERE PHYSICAL OR ENVIRONMENTAL DAMAGE (COLLECTIVELY, "HIGH RISK ACTIVITIES"). PROSOFT SPECIFICALLY DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY OF FITNESS FOR HIGH RISK ACTIVITIES.

9.5.4 DISCLAIMER OF ALL OTHER WARRANTIES

THE WARRANTIES SET FORTH IN PARAGRAPH 1 ABOVE ARE IN LIEU OF ALL OTHER WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

*9.5.5 LIMITATION OF REMEDIES***

IN NO EVENT WILL PROSOFT (OR ITS DEALER) BE LIABLE FOR ANY SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES BASED ON BREACH OF WARRANTY, BREACH OF CONTRACT, NEGLIGENCE, STRICT TORT, OR ANY OTHER LEGAL THEORY. DAMAGES THAT PROSOFT AND ITS DEALER WILL NOT BE RESPONSIBLE FOR INCLUDE, BUT ARE NOT LIMITED TO: LOSS OF PROFITS; LOSS OF SAVINGS OR REVENUE; LOSS OF USE OF THE PRODUCT OR ANY ASSOCIATED EQUIPMENT; LOSS OF DATA; COST OF CAPITAL; COST OF ANY SUBSTITUTE EQUIPMENT, FACILITIES, OR SERVICES; DOWNTIME; THE CLAIMS OF THIRD PARTIES, INCLUDING CUSTOMERS OF THE PURCHASER; AND INJURY TO PROPERTY.

** Some areas do not allow time limitations on an implied warranty, or allow the exclusion or limitation of incidental or consequential damages. In such areas the above limitations may not apply. This Warranty gives you specific legal rights, and you may also have other rights which vary from place to place.

9.5.6 Time Limit for Bringing Suit

Any action for breach of warranty must be commenced within 15 months (or in the case of RadioLinx modules, 39 months) following shipment of the Product.

9.5.7 No Other Warranties

Unless modified in writing and signed by both parties, this Warranty is understood to be the complete and exclusive agreement between the parties, suspending all oral or written prior agreements and all other communications between the parties relating to the subject matter of this Warranty, including statements made by salesperson. No employee of ProSoft or any other party is authorized to make any warranty in addition to those made in this Warranty. The Customer is warned, therefore, to check this Warranty carefully to see that it correctly reflects those terms that are important to the Customer.

9.5.8 Intellectual Property

A Any documentation included with Product purchased from ProSoft is protected by copyright and may not be photocopied or reproduced in any form without prior written consent from ProSoft.

B ProSoft's technical specifications and documentation that are included with the Product are subject to editing and modification without notice.

C Transfer of title shall not operate to convey to Customer any right to make, or have made, any Product supplied by ProSoft.

D Customer is granted no right or license to use any software or other intellectual property in any manner or for any purpose not expressly permitted by any license agreement accompanying such software or other intellectual property.

E Customer agrees that it shall not, and shall not authorize others to, copy software provided by ProSoft (except as expressly permitted in any license agreement accompanying such software); transfer software to a third party separately from the Product; modify, alter, translate, decode, decompile, disassemble, reverse-engineer or otherwise attempt to derive the source code of the software or create derivative works based on the software; export the software or underlying technology in contravention of applicable US and international export laws and regulations; or use the software other than as authorized in connection with use of Product.

9.5.9 Additional Restrictions Relating To Software And Other Intellectual Property

In addition to complying with the Terms of this Warranty, Customers purchasing software or other intellectual property shall comply with any license agreement accompanying such software or other intellectual property. Failure to do so may void this Warranty with respect to such software and/or other intellectual property.

9.5.10 Allocation of risks

This Warranty allocates the risk of product failure between ProSoft and the Customer. This allocation is recognized by both parties and is reflected in the price of the goods. The Customer acknowledges that it has read this Warranty, understands it, and is bound by its Terms.

9.5.11 Controlling Law and Severability

This Warranty shall be governed by and construed in accordance with the laws of the United States and the domestic laws of the State of California, without reference to its conflicts of law provisions. If for any reason a court of competent jurisdiction finds any provisions of this Warranty, or a portion thereof, to be unenforceable, that provision shall be enforced to the maximum extent permissible and the remainder of this Warranty shall remain in full force and effect. Any cause of action with respect to the Product or Services must be instituted in a court of competent jurisdiction in the State of California.

Index

[

[Backplane Configuration] • 63
[MCM Port X] • 65
[Modbus Port 1 Commands] • 70
[Module] • 63

0

00 Return Query Data • 121

3

3x Register Start • 64

4

4x Register Start • 64

A

Add the PTQ Module to the Project • 20, 39
Adding a Module • 60
Adding a Project • 59
Additional Restrictions Relating To Software And Other Intellectual Property • 134
All Product Returns • 129
Allocation of risks • 134
Analyzing Data for the first application port • 87
Analyzing Data for the second application port • 87

B

Backplane Data Transfer • 99, 102
Backplane Menu • 86
Baud Rate • 67
Bit Input Offset • 68
Block Pass-Thru • 108
Block Pass-Thru Control Blocks • 109
Build the Project • 41

C

Cable Connections • 49
Can I use the module in a hot backup system? • 125
Clearing Diagnostic Data • 83
Cmd Err Pointer • 69
Cold Boot • 104
Command Control Block • 101
Command Count • 69
Command List Entry Errors • 95
Comment Entries • 61
Configuration Data • 112
Configuring the Floating Point Data Transfer • 71
Configuring the Processor with Concept • 15
Configuring the Processor with ProWORX • 33
Configuring the Processor with UnityPro XL • 37
Connect the PC to the ProTalk Configuration/Debug Port • 53
Connect Your PC to the Processor • 42
Connecting to the Processor with TCPIP • 44
Controlling Law and Severability • 135
Create a New Project • 17, 37

D

Data Analyzer • 87
Data Analyzer Tips • 89
Data Bits • 67
Diagnostics (Function Code 08) • 119
Diagnostics and Troubleshooting • 9, 77, 79
Direct Pass-Thru • 105
DISCLAIMER OF ALL OTHER WARRANTIES • 133
DISCLAIMER REGARDING HIGH RISK ACTIVITIES • 133
Displaying Timing Marks in the Data Analyzer • 87
Does the module work in a remote rack? • 125
Download the Project to the Module • 76
Download the Project to the Processor • 25, 44

E

Edit the Configuration File • 62
Enable • 65
ENRON Floating Point Support • 71
Error Codes • 95
Error Delay Counter • 70
Error Status Table • 95
Error/Status Block Pointer • 65
Event Command • 100
Example - Function 15 • 109
Example and state diagram • 121
Exiting the Program • 84

F

Failure Flag Count • 64
Features and Benefits • 97
Float Flag • 66
Float Offset • 66
Float Start • 66
Force Multiple Coils (Function Code 15) • 122
Force Single Coil (Function Code 05) • 118
Frequently Asked Questions • 125
Function 15 • 109
Function 5 • 109
Function 6 and 16 • 109
Functional Overview • 9, 99
Functional Specifications • 98

G

General Specifications • 97
Guide to the PTQ-MCM Application Reference Guide • 9

H

Hardware and Software Requirements • 11
Hardware Specifications • 98
Hold Offset • 69

How to Contact Us
Sales and Support • 128

I

Information for Concept Version 2.6 Users • 16
Initialize Output Data • 64
Inserting the 1454-9F connector • 48
Install ProSoft Configuration Builder Software • 13
Install the ProTalk Module in the Quantum Rack • 47, 48
Installing MDC Configuration Files • 16
Intellectual Property • 134
Is a .MDC available for configuration of the Module? • 125

K

Keystrokes • 82

L

LED Status Indicators • 94
LIMITATION OF REMEDIES** • 133
LIMITED WARRANTY • 131

M

Main Menu • 82
Master Command Error List Menu • 93
Min Cmd Delay • 69
Minimum Response Delay • 67
Modbus Database View • 84
MODBUS Exception Codes • 124
MODBUS Exception Responses • 123
Modbus Protocol Specification • 84, 114
Modifying the Configuration File • 57
Module Communication Error Codes • 95
Module Entries • 61
Module Name • 63
Module Type Parameter • 63
Moving Back Through 5 Pages of Registers • 85

N

Navigation • 81
No Other Warranties • 134

O

Opening the Backplane Menu • 83
Opening the Command List Menu • 92
Opening the Database Menu • 83
Opening the Protocol_Serial_ Menu • 83
Opening the Serial Port Menu • 92
Output Offset • 68
Overview • 105

P

Parity • 67
Pass Thru Address • 66
Pass-Thru Configuration • 105
Pass-Thru Control Blocks • 105
PC and PC Software • 12
Please Read This Notice • 2
Port 1 Commands • 113
Port 1 Setup • 112
Port 2 Commands • 114
Port 2 Setup • 113
Port Specific Information • 114
Preset Multiple Registers (Function Code 16) • 122
Preset Single Register (Function Code 06) • 119
Printing a Configuration File • 62
Procedures for Return of Units Out of Warranty • 130
Procedures for Return of Units Under Warranty • 129
Product Specifications • 9, 97
ProSoft Configuration Builder • 57
ProTalk Module Carton Contents • 11
Protocol • 66
Protocol Serial Menu • 91
PTQ Installation and Operating Instructions • 2
Purchasing Warranty Extension • 131

Q

Quantum / Unity Hardware • 12

R

Read Block
Command Control • 104
Event Command • 101
Read Holding Registers (Function Code 03) • 116
Read Input Registers (Function Code 04) • 117
Read Input Status (Function Code 02) • 115
Read Output Status (Function Code 01) • 114
Read Register Count • 64
Read Register Start • 63
Reading Status Data from the Module • 79
Redisplaying the Current Page • 85, 93
Redisplaying the Menu • 86, 91
Reference • 9, 97
Removing Timing Marks in the Data Analyzer • 88
Required Hardware • 79
Response Timeout • 69
Retry Count • 70
Return Material Authorization (RMA) Policies and Conditions • 128
Returning to the Main Menu • 85, 86, 89, 92, 93
RS-232 • 50
Modem Connection • 50
Null Modem Connection (Hardware Handshaking) • 51
Null Modem Connection (No Hardware Handshaking) • 51
RS-232 Configuration/Debug Port • 49, 114
RS-422 • 52
RS-485 • 52
RS-485 and RS-422 Tip • 52
RTS Off • 67
RTS On • 67

S

Serial Port Details • 114
Set Module Parameters • 60
Set up Data Memory in Project • 22

Set Up the Project • 58
Setting Up the ProTalk Module • 47
Skipping 500 Registers of Data • 85
Slave Address • 68
Special Functions • 100
Standard Modbus Protocol Errors • 95
Start Here • 9, 11
Starting the Data Analyzer • 88
Status Data Block Structure • 110
Status Data Definition • 110
Stop Bits • 67
Stopping the Data Analyzer • 88
Sub-function codes supported by the serial line devices • 120
Support, Service & Warranty • 9, 127

T

The Configuration/Debug Menu • 80
Time Limit for Bringing Suit • 133
Transferring the Configuration File from PC to PTQ module • 83
Transferring the Configuration File from PTQ module to PC • 83
Type • 66

U

Un-repairable Units • 130
Use CTS Line • 68
Using the Diagnostic Window in ProSoft Configuration Builder • 80

V

Verification and Troubleshooting • 9, 77
Verify Jumper Settings • 47
Verify Successful Download • 28
Viewing Backplane Diagnostic Information • 86
Viewing Configuration Information • 86, 92
Viewing Data in ASCII (Text) Format • 85, 88
Viewing Data in Decimal Format • 85
Viewing Data in Floating Point Format • 85
Viewing Data in Hexadecimal Format • 85, 88
Viewing Error and Status Data • 92
Viewing Register Pages • 85
Viewing the Next 100 Registers of Data • 85
Viewing the Next 20 Commands • 93
Viewing the Next Page of Commands • 93
Viewing the Previous 100 Registers of Data • 85
Viewing the Previous 20 Commands • 93
Viewing the Previous Page of Commands • 93
Viewing Version Information • 82, 86, 92

W

Warm Boot • 104
Warm Booting the Module • 84
What Is Covered By This Warranty • 131
What Is Not Covered By This Warranty • 132
What kind of data transfer rates can I expect between the PLC and the module? • 125
Word Input Offset • 68
Write Block - Command Control • 103
Write Block - Event Command • 100
Write Register Count • 64
Write Register Start • 64

Y

Your Feedback Please • 3